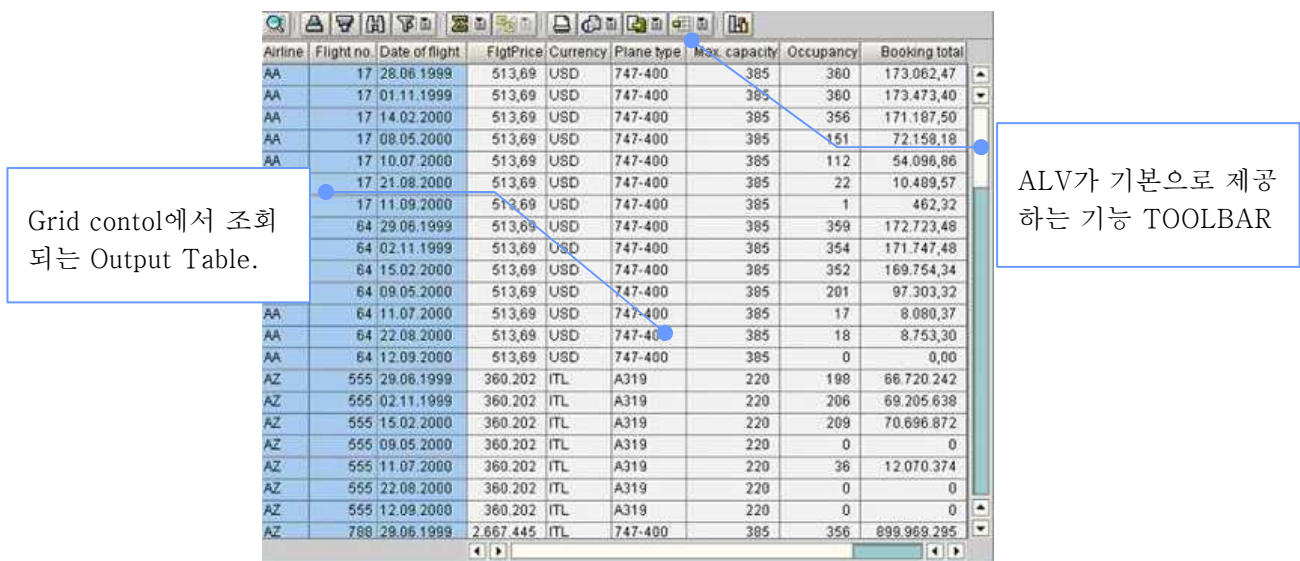


## 1. OVERVIEW

## 1-1. What is ALV?

ALV는 Abap List Viewer의 약자로서 리스트 화면에 편리한 기능을 제공해준다. 15단원의 레포트 프로그램에서는 write 구문으로 Output 화면을 구성하며, 조회된 데이터를 엑셀로 다운로드 받는 등의 기능은 버튼을 추가하여 이벤트 처리를 해주어야 한다. 그러나 ALV는 이러한 공통적인 사무작업들이 기본적으로 포함되어 있다. 엑셀 프로그램에서 정렬하고, 합계를 구하는 등의 기본적인 작업들을 ALV에서 최종 사용자가 자유롭게 사용할 수 있게 해준다. SAP release 3.1 버전 이후 ALV가 프로그램에서 등장하게 되었으며, 보고서나 성적서와 같이 포맷이 일정한 문서 이외의 데이터 조회는 대부분 ALV로 개발되고 있다. ALV도 크게 함수를 이용하는 방법과 GRID 컨트롤을 이용하는 방법이 있는데, 이 두 가지 방식은 내부적으로 유사한 구조를 가지며 유사한 기능을 가지고 있다. 함수를 이용하여 ALV를 호출하는 것은 REUSE\_ALV\_GRID\_DISPLAY 함수가 자동으로 스크린을 생성하여 리스트를 화면에 보여주게 된다.

이번 단원에서는 GRID를 사용하는 ALV를 학습하게 된다. GRID는 클래스로 개발된 기술이므로 17단원의 ABAP 오브젝트를 이해하고 있어야만 쉽게 접근할 수 있다. [그림 18-1]은 ALV로 리스트를 조회한 화면이다. ALV GRID 컨트롤은 화면 Display에 관련되어서 이미 개발된 Contol 기술을 사용한다. 다른 모든 Contol과 동일하게, ALV Grid 컨트롤은 Global 클래스를 통해 속성에 영향을 미칠 수 있는 메소드들을 제공하고 있다. ABAP Objects를 사용한 결과로서, ALV 인스턴스를 통해 리스트가 보여지고, 프로그래머들은 ABAP Objects의 이벤트 기술을 사용하여야 한다. Control은 개인(Local) PC에 설치된 소프트웨어 컴포넌트이다. 이러한 컴포넌트들과 통신하기 위해서는, 모든 Control은 *Control Framework*의 메소드를 이용하여야 한다. ALV Grid Control은 필요한 메소드들을 이미 포함하고 있는 특별한 경우이다. 이것은 프로그래머에게 Control의 이벤트 관리 프로세스 단계를 건너뛸 수 있는 효율성을 제공해준다. ALV Grid Control은 표준 기능들을 통합하기 위해 SAP context 메뉴를 사용하고 있다. 이러한 메뉴들은 프로그램 내에서 필요여부에 따라 추가/삭제가 가능하다.



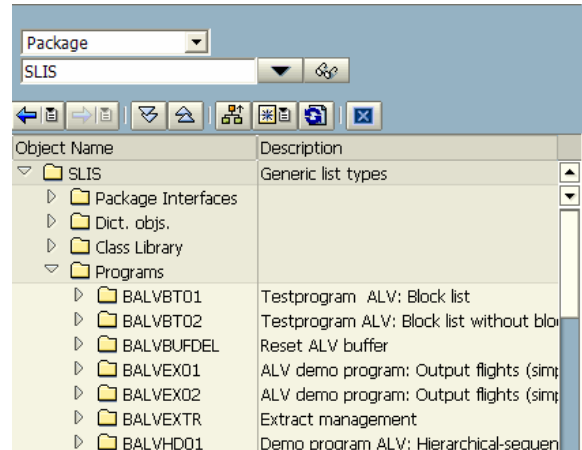
▲ 그림 18-1. ALV(Abap List Viewer)

## ALV에서 제공하는 기능

- 비계층구조 및 계층구조 조회 가능
- 전형적인 리스트 기능을 사용  
(추가적인 소스코드 없이 소트, 필터등의 기능사용)
- 이미정의되어 있는 기능과 확장기능을 선택 적용
- 사용자 액션(더클릭과 같은)에 독립적으로 반응
- REPORT로 연결될수 있다.

ALV 다양한 예제는 SLIS 패키지에 등록되어 있으므로 참고한다.

(단, 함수를 이용한 ALV도 존재하므로 구분하기바란다.)



▲ 그림 18-2. SLIS 패키지

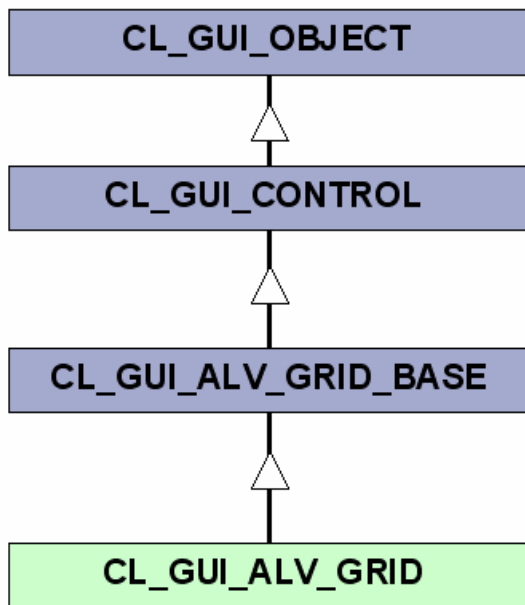
세부적으로 다음과 같은 표준 기능을 제공한다.

- |  |  |
|--|--|
| a) Navigating Within the List.                     | l) Displaying the List Status.                 |
| b) Choosing Detail.                                | m) Optimizing the Column Width.                |
| c) Sorting in Ascending/Descending Order.          | n) Freezing to Columns and Unfreezing Columns. |
| d) ABC Analysis.                                   | o) Choosing Display Variants.                  |
| e) Selecting and Deselecting Rows.                 | p) Defining the Current Display Variant.       |
| f) Row indicator                                   | q) Saving Display Variants.                    |
| g) Setting and Deleting Filters.                   | r) Management of Display Variants.             |
| h) Displaying and Deleting Sums.                   | s) Displaying The Basic List.                  |
| i) Creating Subtotals.                             | u) Finding Terms.                              |
| j) Choosing Summation Levels.                      | v) Printing Lists.                             |
| k) Defining the Breakdown of the Summation Levels. | w) Sending Lists as Documents                  |
|  | x) SpreadSheets.                               |
|  | y) Word Processing.                            |
|  | z) Local File                                  |

## 1-2. Instance for the ALV Grid Control

ALV 프로프로그램에서 사용하게 될 인스턴스는 CL\_GUI\_ALV\_GRID 클래스의 참고하는 변수로 정의되어 있다. 다음 구문을 이용하여 객체 참고 변수를 선언한후 인스턴스를 생성하게 된다. 바로 본문으로 들어가므로 이해가 되지 않더라도 진행하자.

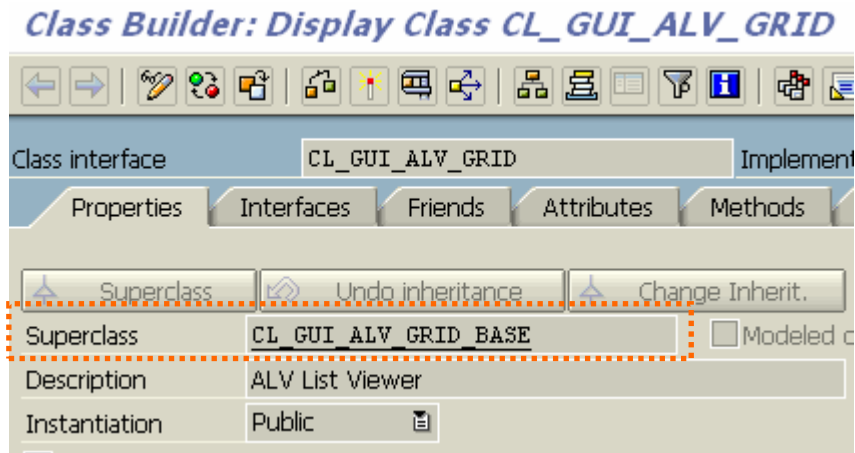
DATA : name TYPE RET TO CL\_GUI\_ALV\_GRID.



▲ 그림 18-3. ALV의 상속트리

ALV Grid의 인스턴스는 CL\_GUI\_ALV\_GRID를 참고하는 Reference 변수이다. ALV Grid 컨트롤은 화면에 보여지는 모든 정보를 가지고 있다. 메소드를 호출하여 화면의 속성을 재정의하고 변경할수 있다.

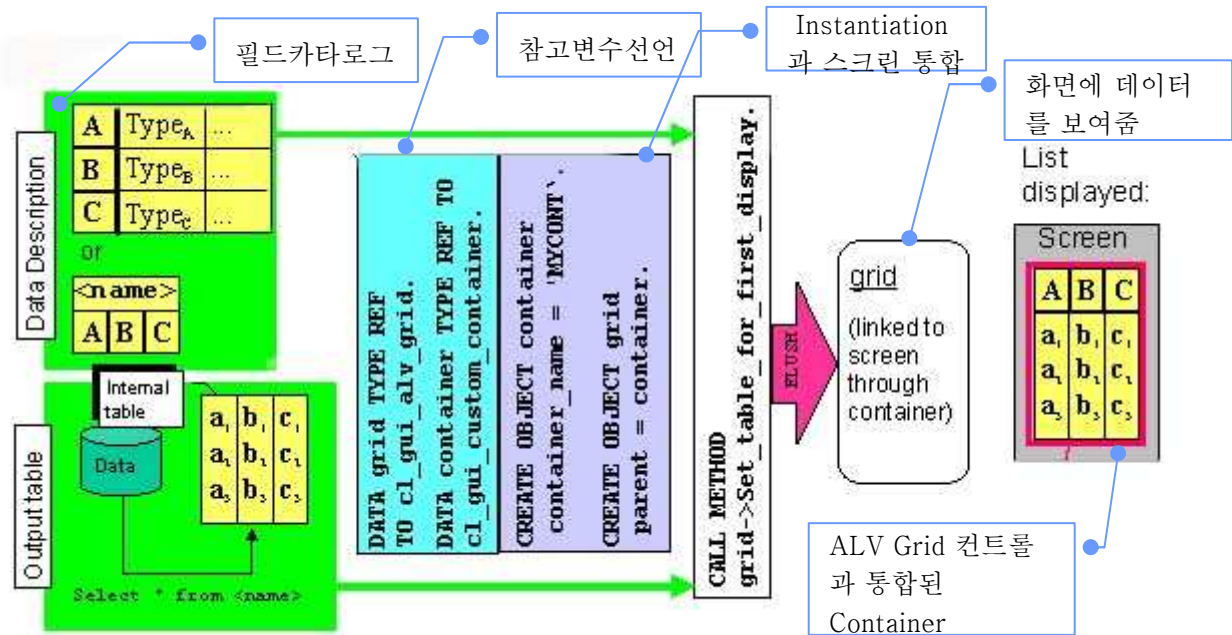
[그림 18-3]은 ALV의 상속트리를 보여주고 있다. SE43 클래스 빌더에서 CL\_GUI\_ALV\_GRID를 조회해보면 슈퍼클래스필드에 상의의 클래스가 존재한다.



▲ 그림 18-4. ALV의 클래스 조회

## 1-2. Working With the ALV Grid Control

[그림 18-4]는 ALV GRID가 화면에 보여지는 순서를 설명하고 있다.



▲ 그림 18-4. ALV의 실행 순서

ALV를 이용하여 데이터를 화면에 뿌려주기 위해서는 최소한 다음 2가지 작업을 해주어야 한다.

- **인터널 테이블 선언** : 화면에 보여지게 될 데이터
- **데이터의 구조(필드카타로그)** : Output table이라고 불리는 테이블 구조를 정의함 – ALV Grid 컨트롤에서 정의되는 데이터의 구조 내역 정보를가지고 있다. 일반적으로 인터널 테이블의 구조를 그대로 사용하거나, ABAP Dictionary의 구조체를 이용한다.

ALV Grid 컨트롤에 넘겨지게되는 Output table에 대한 참고는 ALV Grid 컨트롤이 작동하고 있는 이상은 유용하게 된다. 필드카타로그(Field catalog)는 ALV 화면에 보여지게 되는 필드들의 정보를 담고 있는 테이블이다. 예를들어, ALV 필터의 타입, 속성, 길이등을 정의할게 된다. 필드카타로그는

LVC\_T\_FCAT 타입의 테이블이다. 뒤에서 좀더 자세하게 살펴보도록 하자.

ALV는 화면의 SAP Container와 연결되어 화면에 뿌려지게 된다. Container라는 단어의 의미에서 알 수 있듯이 화면내에서 “무엇을 담는다” 는 것을 표한한다. 컨테이너네에는 Textedit Control, Picture Control과 같은 Control 오브젝트를 화면에 보여줄때 사용하게 되는 상위의 Control이다.

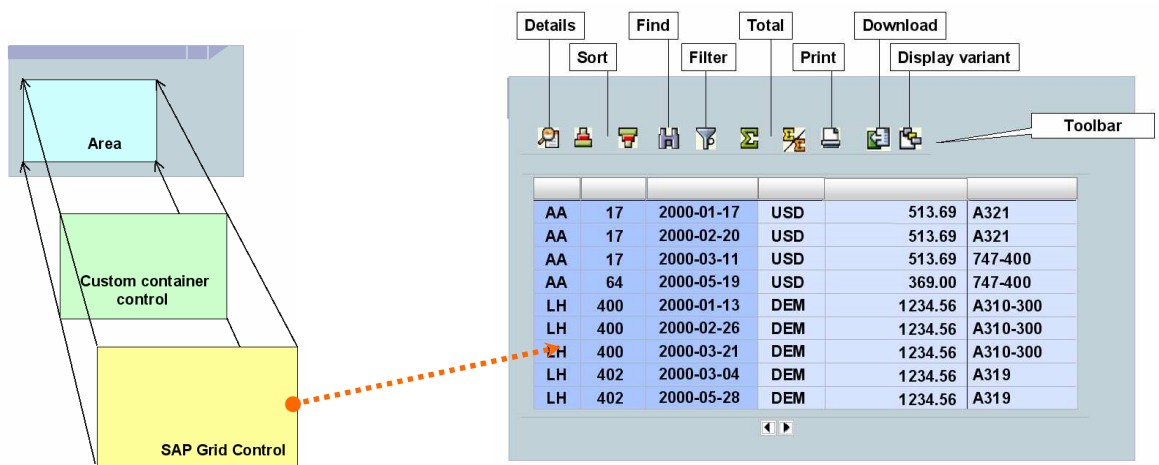
## 2. SAP Container

스크린에 ALV 인스턴스를 물리적으로 화면에 보이게 하기 위해서는 연결고리 역할을 하는 SAP Container Control이 반드시 존재하여야 한다. 즉, Sap Container는 linker로서의 연결을 하기위해 Sap Control들을 자기 영역안에 포함하는 컨테이너 역할을 하게 된다. Sap control의 종류로는 SAP Tree Control, SAP Picture Control, SAP Textedit Control, SAP Splitter Control등이 있다.

SAP Container도 다른 control을 포함하는 control의 하나이며, Parent Control이라고도 한다. 이러한 컨트롤들을 화면에 보이게 하기 위한 SAP Container는 5개의 종류가 있다.

| 컨테이너의 종류                    | 기능  |
|-----------------------------|---|
| SAP Custom Container        | 스크린 페인터를 사용하는 일반적인 화면에서 영역을 정의하게 된다.<br>Class: CL_GUI_CUSTOM_CONTAINER                            |
| SAP Dialog Box Container    | Dialog Box 또는 Fullscreen에서 Dialog Box 형태로 보이도록 한다.<br>Class: CL_GUI_DIALOGBOX_CONTAINER           |
| SAP Docking Container       | 스크린 영역의 각 모서리에 붙어서 사이즈를 조절할 수 있게 한다.<br>Class: CL_GUI_DOCKING_CONTAINER                           |
| SAP Splitter Container      | 여러 영역으로 Container를 분리할때 사용된다<br>Class: CL_GUI_SPLITTER_CONTAINER .                                |
| SAP Easy Splitter Container | Splitter 컨트롤과 비슷한 역할을 수행하며, 분리된 영역을 상하 좌우로 한번더 분리할 수 있다.<br>Class: CL_GUI_EASY_SPLITTER_CONTAINER |

▲ 표 18-2-1. SAP Container의 종류

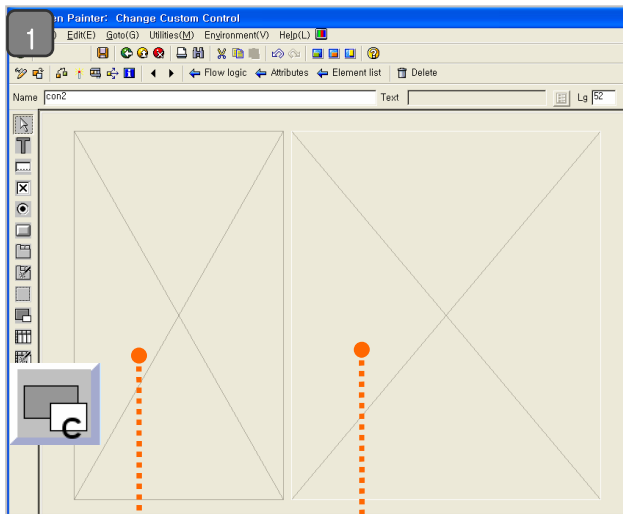


▲ 그림 18-2-1. SAP Container와 ALV

## 2-1. SAP Custom Container

### 2-2-1 SAP Custom Container 생성

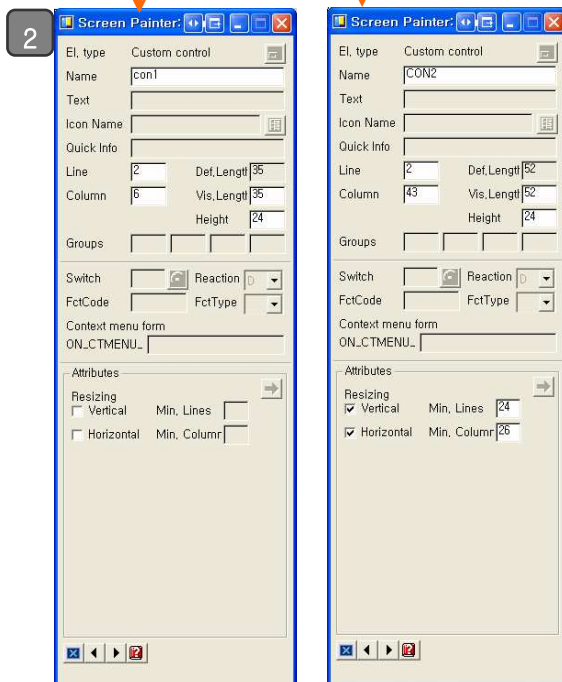
5가지의 Container중에서 SAP Custom Container에 대해서만 학습하자. 이외의 것은 기본원리는 비슷하며 사용법만 익히면 어려움없이 적용할수 있다. 지금 하고 있는 작업은 ALV를 화면에 보여주기 위해 화면에서 영역을 지정하고, 이 영역에 ALV를 올리기 위한 작업장을 만드는 것이다. 프로그램 Z18\_001을 생성하여 스킨 100번을 추가하자.



1. 스크린 페인터를 이용하여 Custom Control 생성  
스크린페인터를 오픈하여 Custom Control 버튼을 클릭하면 화면에 영역을 지정할 수 있다. 2개의 영역을 drag & drop으로 적절한 크기로 생성하자.

#### 2. Custom Control 이름 및 속성 지정

컨테이너를 더블클릭하면 스크린 페인터 속성창이 열린다. Con2에는 Resizing 속성을 체크하자



| 컨테이너 속성                             | 기능   |
|-------------------------------------|--|
| Resizing<br>Vertical/<br>horizontal | 윈도우의 창 크기에 따라 화면에서 차지하는 영역을 비율로 조절하여 보여준다. |
| Min. lines<br>Min. columns          | 화면에 보여지게 될 최소한의 라인과 컬럼수를 설정한다.             |

▲ 표 18-2-2. SAP Custom Container의 속성

## 2-2-2 Custom Container 오브젝트 생성

5가지의 Container중에서 SAP Custom Container에 대해서만 학습하자. 이외의 것은 기본원리는 비슷하며 사용법만 익히면 어려움없이 적용할수 있다. 지금 하고 있는 작업은 ALV를 화면에 보여주기 위해 화면에서 영역을 지정하고, 이 영역에 ALV를 올리기 위한 작업장을 만드는 것이다. 프로그램 Z18\_001을 생성하여 스킨 100번을 추가하자.

```
3 REPORT Z18_001.

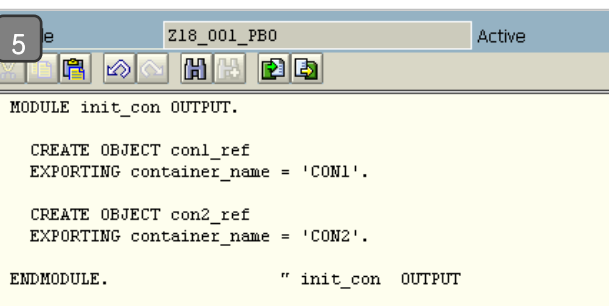
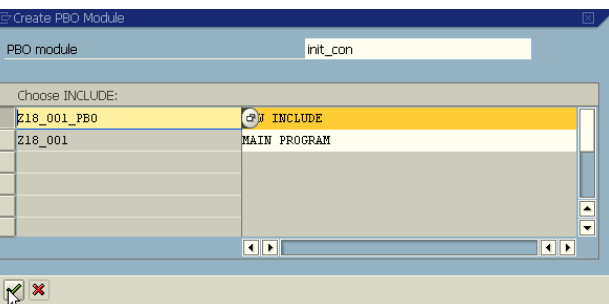
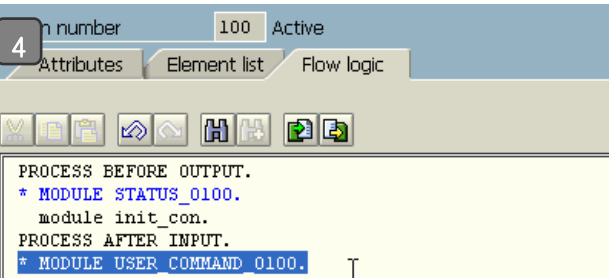
data : con1_ref type ref to
      CL_GUI_CUSTOM_CONTAINER,
      con2_ref type ref to
      CL_GUI_CUSTOM_CONTAINER.
```

### 3. 컨테이너 참고 변수 생성

REPORT 프로그램 상단에 CON1, CON2라는 객체 참고 변수를 생성하자.

### 4. PBO 생성

스크린 100번에 SAP Customer Container 오브젝트를 생성하기 위해 PBO 모듈을 생성하자. INCLUDE 이름은 Z18\_001\_PBO로 한다.



### 5. 컨테이너 오브젝트 생성

CREATE OBJECT 구문으로 컨테이너 오브젝트를 생성하면서, 컨테이너 이름은 'CON1'로 지정한다. 이것은 컨테이너 오브젝트와 스크린의 SAP Custom 컨테이너를 연결하는 작업이다.

| 컨테이너 속성                     | 기능   |
|-----------------------------|--|
| parent                      | 컨트롤이 보여지는 인스턴스의 상위 컨트롤 지정  |
| container_name              | 스크린 페인터에서 지정한 Custom Container 이름 지정                               |
| style                       | 컨트롤의 외형적 stype 지정  |
| dynnr                       | 컨트롤에 추가하고자 하는 스크린 번호   |
| repid                       | 컨트롤에 추가하고자 하는 프로그램 ID  |
| lifetime                    | 컨트롤의 생명주기 설정 (LEAVE TRANSACTION, CALL TRANSACTION등의 명령에 따라 비활성 설정) |
| no_autodef_pro<br>gid_dynnr | 프로그램 ID와 스크린 번호 자동지정('X'를 설정하면 OFF)                                |

▲ 표 18-2-3. SAP Custom Container 파라미터 속성



6 REPORT Z18\_001.

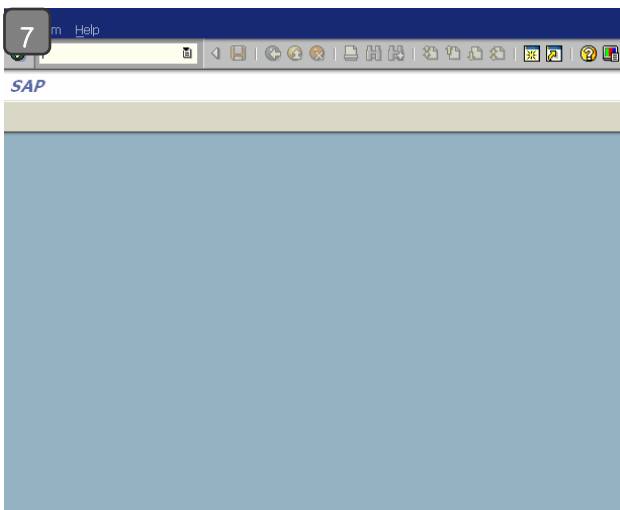
```
data : con1_ref type ref to  
      CL_GUI_CUSTOM_CONTAINER,  
      con2_ref type ref to  
      CL_GUI_CUSTOM_CONTAINER.
```

```
call screen 100.
```

```
INCLUDE Z18_001_PB0.
```

## 6. 화면 호출

100번 스크린에서 Customer Contol을 생성하였으므로 100번 화면을 호출하는 스크립트를 추가한다. Include 구문은 4번 단계에서 생성하였으므로 자동으로 추가된다.



## 7. 프로그램 실행

프로그램을 실행하게 되면 화면에는 화면에는 아무것도 보이지 않는다. 지금까지 작업한 것은 화면에 ALV와 같은 컨트롤 올리기 위한 AREA 를 지정한 것이다.

## 8. ALV 생성

화면이 어떻게 구성되는지 살펴보기 위해 간단히 AVL GRID를 생성하자. GRID1, GRID2를 메인 프로그램 데이터 선언부분에 추가한다.

## 9. ALV 오브젝트 생성

4번단계의 INIT\_CON에 다음 소스를 추가한다.

8 REPORT Z18\_001.

```
data : con1_ref type ref to  
      CL_GUI_CUSTOM_CONTAINER,  
      con2_ref type ref to  
      CL_GUI_CUSTOM_CONTAINER.
```

```
DATA : grid1 type ref to cl_gui_alv_grid,  
      grid2 type ref to cl_gui_alv_grid.
```

```
create object grid1  
  EXPORTING  
    i_parent = con1_ref.
```

```
create object grid2  
  EXPORTING  
    i_parent = con2_ref.
```



## 10. 프로그램 재실행

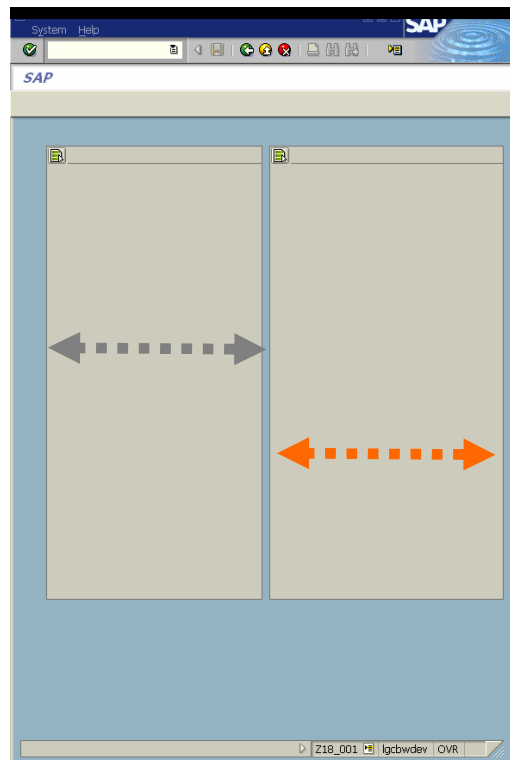
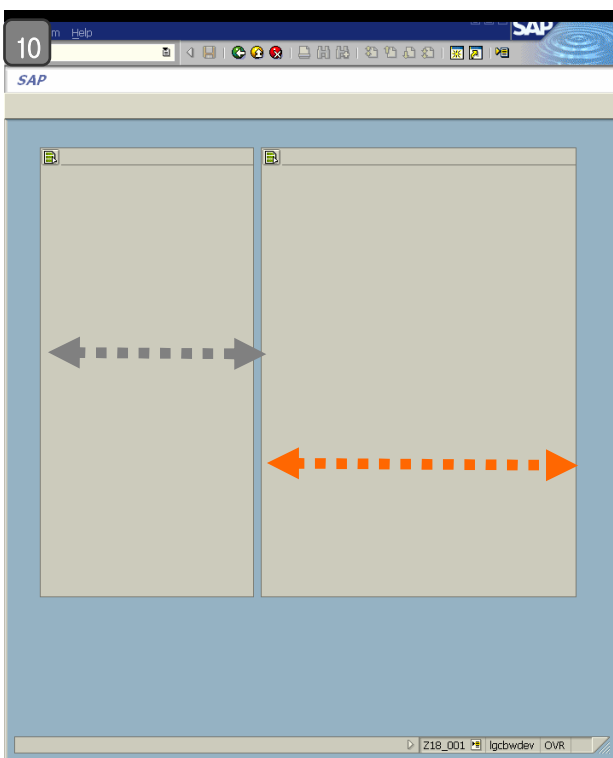
ALV GRID를 컨테이너에 올려서 프로그램을 재실행한 결과이다.

2번단계에서 RESIZING 속성을 설정 한 것을 확인해보자. 윈도우 창 크기를 조절해보면 컨테이너 2번은 가로 사이즈가 줄어드는 반면에 컨테이너 1번은 아무런 차이가 없음을 알수 있다.

앞의 과정을 요약하면

- 1). 스크린에서 Customer Control을 생성하였다.(con1).
- 2). SAP Continaer 참고 변수를 생성하였다.(con1\_ref)
- 3). SAP Container 오브젝트를 스크린의 Customer Control과 연결하여 생성한다.
- 4). ALV GRID 참고 변수를 생성하여 SAP Container위에 올린다.

이러한 과정을 진행하면 10번 화면과 같은 결과를 얻게 된다. 현재 보이는 것은 단순한 ALV GRID 이며, ALV가 EXEL과 같은 시트 모양을 가질수 있도록 컬럼을 추가하고 데이터를 추가하는 과정을 다음 장에서 살펴보도록 하자. 해당 소스는 Z18\_001, Z18\_001\_PBO 로 소스 자료실에 올려두었으므로 참고 하기 바란다.



### 3. ALV 생성하기

2장에서 스크린에 Container를 생성하여 ALV를 추가하는 실습을 하였다. 이때 화면에 보여지는 것은 단순히 ALV의 구조와 데이터가 보여지게 될 공간을 만든 것 이므로, 이번장에서는 ALV를 생성하여 화면에 데이터를 보여지는 것을 실습해보자.

첫째 화면에 추가할 ALV Grid 컨트롤의 인스턴스를 생성하자.

둘째, 인스턴스의 필드를 생성하고 화면에 보여줄 데이터를 select 하자.

셋째, set\_table\_for\_first\_display 메소드를 호출하여 화면에 보여주자.

#### 3-1. ALV Grid Control 생성하기

##### 1. 객체변수 생성

ALV Grid 컨트롤을 참고하는 객체 변수를 생성하자. 이와 동시에 화면에 보여줄 인터널 테이블도 생성하자. 그리고 스크린 100번을 생성하고 Customer control 'CON1'을 추가하자.

1 REPORT Z18\_002.

data : con1\_ref type ref to CL\_GUI\_CUSTOM\_CONTAINER.

DATA : grid1 type ref to cl\_gui\_alv\_grid.

data : gt\_sflight type table of sflight.

2 MODULE init\_con OUTPUT.

IF con1\_ref IS INITIAL.

CREATE OBJECT con1\_ref

EXPORTING container\_name = 'CON1'.

create object grid1

EXPORTING

i\_parent = con1\_ref.

ENDIF.

ENDMODULE.

3 start-of-selection.

SELECT \* FROM sflight INTO TABLE

gt\_sflight.

call screen 100.

##### 2. 오브젝트 생성

스크린 100번의 PBO 모듈의 init\_con에 다음 코드를 추가하자. IF ~ is initial. 구문은 오브젝트가 한번 생성되었다면 다시 생성하지 않는다는 것이므로, 실제업무에서는 잊지 말고 사용하여야 한다.

##### 3. 데이터 SELECT

메인 프로그램에서 START-OF-SELECTION을 추가하여, 데이터를 SELECT 하자. 그리고 화면 100번을 CALL 한다.

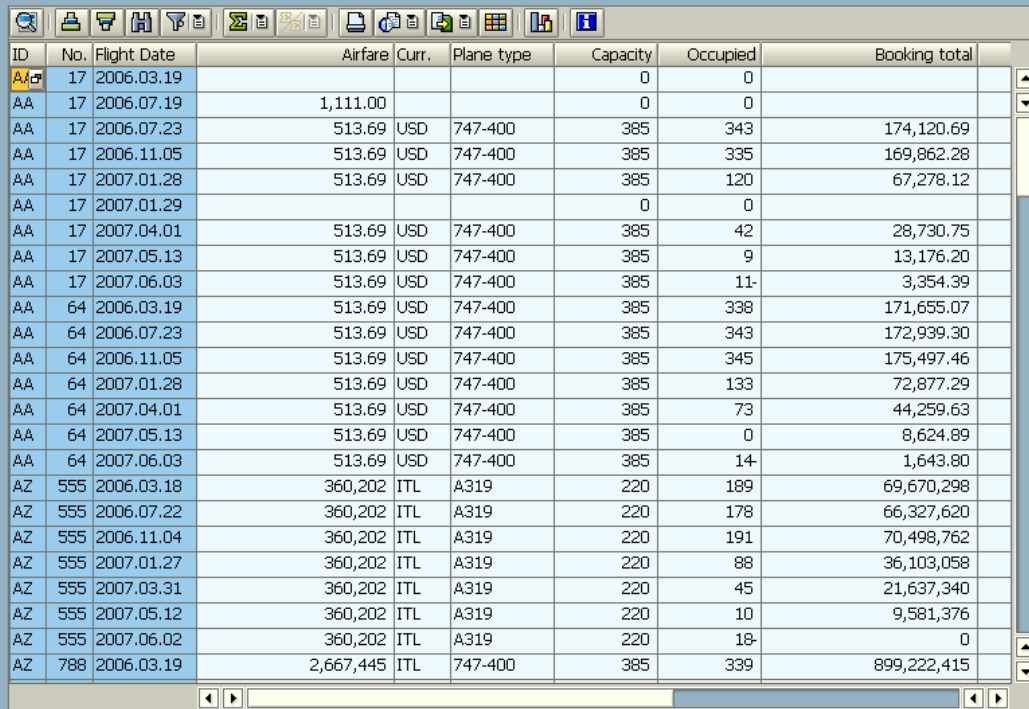
##### 4. ALV display 메소드호출

ALV 를 조회하는 GRID1 인스턴스의 메소드를 호출하기만 하면 ALV 작업은 간단하게 끝낼수 있다. 이때 파라미터는 SFLIGHT 구조체를 이용하고, 화면에 보일 데이터는 GT\_SFLIGHT 이다.

4 CALL METHOD grid1->set\_table\_for\_first\_display

EXPORTING I\_STRUCTURE\_NAME = 'SFLIGHT'

CHANGING IT\_OUTTAB = gt\_sflight.



| ID | No. | Flight Date | Airfare   | Curr. | Plane type | Capacity | Occupied | Booking total |
|----|-----|-------------|-----------|-------|------------|----------|----------|---------------|
| AA | 17  | 2006.03.19  |           |       |            | 0        | 0        |               |
| AA | 17  | 2006.07.19  | 1,111.00  |       |            | 0        | 0        |               |
| AA | 17  | 2006.07.23  | 513.69    | USD   | 747-400    | 385      | 343      | 174,120.69    |
| AA | 17  | 2006.11.05  | 513.69    | USD   | 747-400    | 385      | 335      | 169,862.28    |
| AA | 17  | 2007.01.28  | 513.69    | USD   | 747-400    | 385      | 120      | 67,278.12     |
| AA | 17  | 2007.01.29  |           |       |            | 0        | 0        |               |
| AA | 17  | 2007.04.01  | 513.69    | USD   | 747-400    | 385      | 42       | 28,730.75     |
| AA | 17  | 2007.05.13  | 513.69    | USD   | 747-400    | 385      | 9        | 13,176.20     |
| AA | 17  | 2007.06.03  | 513.69    | USD   | 747-400    | 385      | 11       | 3,354.39      |
| AA | 64  | 2006.03.19  | 513.69    | USD   | 747-400    | 385      | 338      | 171,655.07    |
| AA | 64  | 2006.07.23  | 513.69    | USD   | 747-400    | 385      | 343      | 172,939.30    |
| AA | 64  | 2006.11.05  | 513.69    | USD   | 747-400    | 385      | 345      | 175,497.46    |
| AA | 64  | 2007.01.28  | 513.69    | USD   | 747-400    | 385      | 133      | 72,877.29     |
| AA | 64  | 2007.04.01  | 513.69    | USD   | 747-400    | 385      | 73       | 44,259.63     |
| AA | 64  | 2007.05.13  | 513.69    | USD   | 747-400    | 385      | 0        | 8,624.89      |
| AA | 64  | 2007.06.03  | 513.69    | USD   | 747-400    | 385      | 14       | 1,643.80      |
| AZ | 555 | 2006.03.18  | 360,202   | ITL   | A319       | 220      | 189      | 69,670,298    |
| AZ | 555 | 2006.07.22  | 360,202   | ITL   | A319       | 220      | 178      | 66,327,620    |
| AZ | 555 | 2006.11.04  | 360,202   | ITL   | A319       | 220      | 191      | 70,498,762    |
| AZ | 555 | 2007.01.27  | 360,202   | ITL   | A319       | 220      | 88       | 36,103,058    |
| AZ | 555 | 2007.03.31  | 360,202   | ITL   | A319       | 220      | 45       | 21,637,340    |
| AZ | 555 | 2007.05.12  | 360,202   | ITL   | A319       | 220      | 10       | 9,581,376     |
| AZ | 555 | 2007.06.02  | 360,202   | ITL   | A319       | 220      | 18       | 0             |
| AZ | 788 | 2006.03.19  | 2,667,445 | ITL   | 747-400    | 385      | 339      | 899,222,415   |

## 5. 프로그램 실행

프로그램을 실행하면 자동으로 위와 같은 화면을 보여준다. 너무 간단하지 않은가? 단순히 레포트 기능이라면 이 소스 만으로도 충분히 그럴싸한 화면을 만들 수 있다.

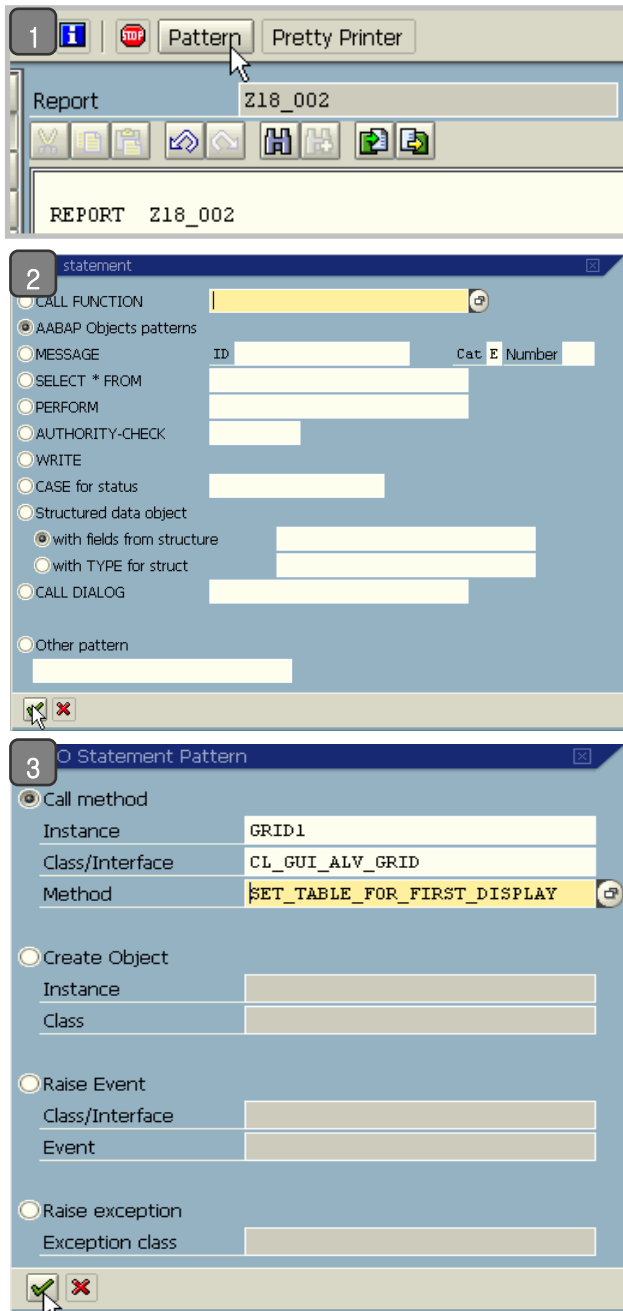
그러나 현업의 요구사항은 단순한 레포트 프로그램이 아니라, ALV에 조회된 데이터도 변경하고 다른 트랜잭션과 연결하는 등의 많은 추가적인 작업이 생겨나게 된다.

다음장부터는 ALV에 사용되는 메소드, 이벤트, 속성(필트카타로그, LAYOUT)에 대해서 구체적으로 살펴보자.

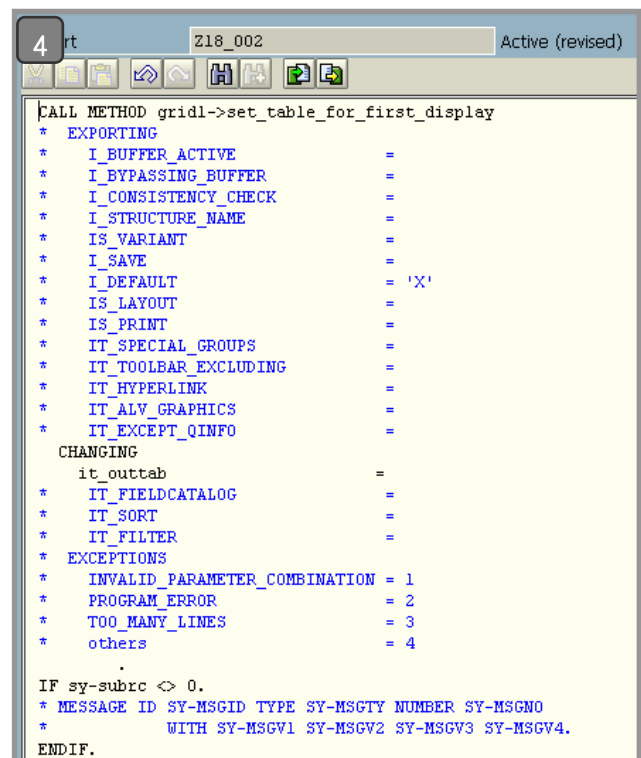
INCLUDE를 메인 프로그램에 합쳐서 Z128\_002 이라는 이름으로 ABAP SOURCE 자료실에 올려두었으니 반드시 프로그램으로 실행해서 확인하기 바란다.

## PATTERN 사용

클래스의 메소드를 호출할 경우에는 파라미터가 많이 존재하고, 파라미터 이름이 길기때문에 직접 입력하기에는 어려움이 따른다. ABAP 에디터의 PATTERN 기능을 이용하여 프로그램 하는 것은 좋은 습관이다.



1. ABAP 에디터의 상단 메뉴 PATTERN 버튼을 클릭한다.
2. APBA Object patterns를 선택하고 엔터를 입력한다.
3. CALL METHOD 를 선택하고 인스턴스명, 클래스명, 메소드명을 입력하고 엔터를 입력한다. 메소드는 Possible entry를 이용하여 검색한다.
4. 인스턴스의 메소드 호출과 파라미터에 대한 스크립트가 화면에 자동으로 입력된다.



## 4. ALV 메소드

메소드는 객체의 행위를 수행한다. ALV 메소드는 ALV의 행위(ALV 조회, ALV Refresh, ALV Sort등)를 관할하게 된다. ALV에 관련된 메소드를 하나하나 살펴보자.

### 4-1. set\_table\_for\_first\_display

3장에서 보았듯이 ALV Control 인스턴스를 OUTPUT 테이블에 조회되게 하는 가장 기본적인 메소드이다. 메소드를 호출할때는 DATA Dictionary의 구조를 참고하거나 **필드카타로그**를 반영하여야 한다. 전자는 앞의 예에서 EXPORTING I\_STRUCTURE\_NAME = 'SFLIGHT' 와 같이 SFLIGHT 테이블과 같은 구조를 참고하는 것을 의미하며, 후자는 직접 ALV의 필드들을 구성하여야 한다는 것을 의미한다. 프로그램을 실행하기 이전에 테이블을 SORT 하거나 필터링 할수 있는 기능을 사용할수 있다.

```
CALL METHOD < ref.var. to CL_GUI_ALV_GRID>->set_table_for_first_display
EXPORTING
    I_STRUCTURE_NAME      = < string of type DD02L-TABNAME>
    IS_VARIANT            = < structure of type DISVARIANT>
    I_SAVE                = < var. of type CHAR01>
    I_DEFAULT             = < var. of type CHAR01>
    IS_LAYOUT             = < structure of type LVC_S_LAYO>
    IS_PRINT              = < structure of type LVC_S_PRNT>
    IT_SPECIAL_GROUPS     = < internal table of type LVC_T_SGRP>
    IT_TOOLBAR_EXCLUDING = < internal table of type UI_FUNCTIONS>
CHANGING
    IT_OUTTAB            = < internal table>
    IT_FIELDCATALOG      = < internal table of type LVC_T_FCAT>
    IT_SORT              = < internal table of type LVC_T_SORT>
    IT_FILTER            = < internal table of type LVC_T_FILT>
```

#### 4-1-1 I\_STRUCTURE\_NAME 파라미터

OUTPUT 테이블의 형태를 만들기 위해 SFLIGHT와 같은 DDIC 구조체 이름을 입력한다. 이 파라미터를 설정하게 되면 필드카타로는 DDIC 구조체에 맞게 자동으로 생성된다. 즉, 필드카타로그를 따로 구성할 필요가 없다.

```
DATA: GRID1 TYPE REF TO CL_GUI_ALV_GRID,
      GT_SFLIGHT TYPE TABLE OF SFLIGHT.

CALL METHOD GRID1->SET_TABLE_FOR_FIRST_DISPLAY
EXPORTING I_STRUCTURE_NAME = 'SFLIGHT'
CHANGING IT_OUTTAB = GT_SFLIGHT.
```

## 4-1-2. IS\_VARIANT 파라미터

ALV 리스트 변형을 설정할수 있다. REPORT 프로그램에서 layout을 설정할수 있도록 설정해야 한다. 리스트 변형은 조회된 화면에서 필드의 순서를 변경하고, 소트를 하는 등의 일련의 작업을 하나의 variant로 저장하여 다음 조회시에도 동일한 포맷으로 조회될수 있도록 하는것이다. Z18\_002 프로그램을 복사생성하여 Variant 기능을 추가해보자. 기존 프로그램 소스에 BOLD채의 소스만 추가하자.

```
REPORT Z18_003 .
```

```
DATA : con1_ref TYPE REF TO  
cl_gui_custom_container.
```

```
DATA : grid1 TYPE REF TO cl_gui_alv_grid,  
      gs_variant LIKE disvariant,  
      gt_sflight TYPE TABLE OF sflight.
```

```
INCLUDE z18_003_pbo.
```

```
INCLUDE z18_003_pai.
```

```
START-OF-SELECTION.
```

```
SELECT * FROM sflight INTO TABLE gt_sflight.  
CALL SCREEN 100.
```

```
DATA: GRID1 TYPE REF TO CL_GUI_ALV_GRID,  
      GT_SFLIGHT TYPE TABLE OF SFLIGHT.
```

```
gs_variant-report = sy-repid.
```

```
gs_variant-username = sy-uname.
```

```
CALL METHOD
```

```
GRID1->SET_TABLE_FOR_FIRST_DISPLAY
```

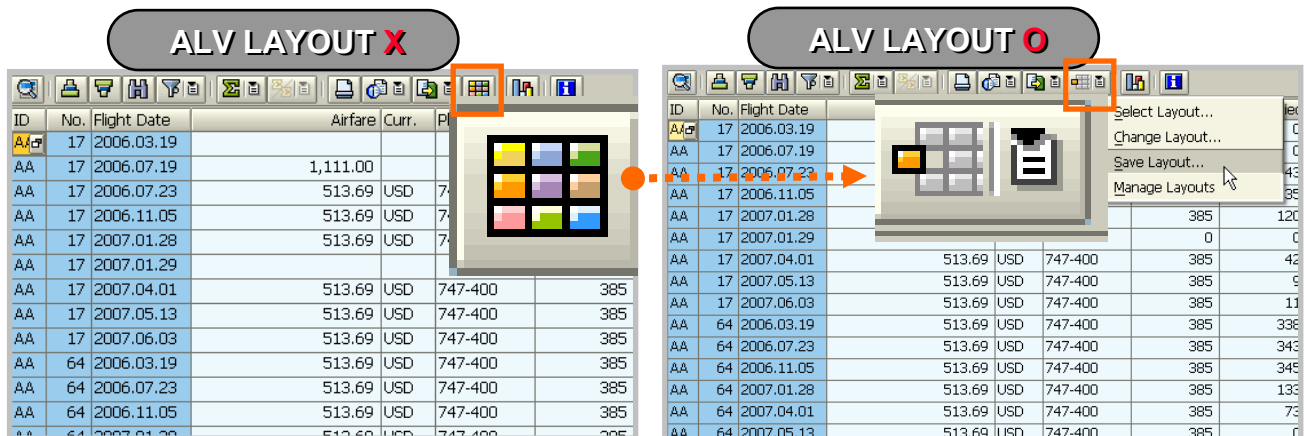
```
EXPORTING
```

```
i_structure_name = 'SFLIGHT'
```

```
is_variant = gs_variant "
```

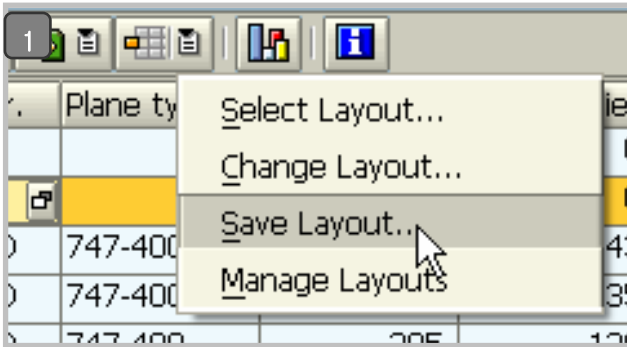
```
i_save = 'A ' ' "
```

```
CHANGING IT_OUTTAB = GT_SFLIGHT.
```



▲ 그림 18-4-1. ALV LAYOUT

[그림 18-4-1]에서 ALV layout 속성을 지정하게 되면 왼쪽의 버튼이 오른쪽 버튼과 같이 LAYOUT을 변경하고 저장할수 있도록 변경된다. 프로그램을 실행하여 필드순서를 변경, 소트, 필터링등을 실행하여 레이아웃을 저장해보면 쉽게 이해할수 있을 것이다.

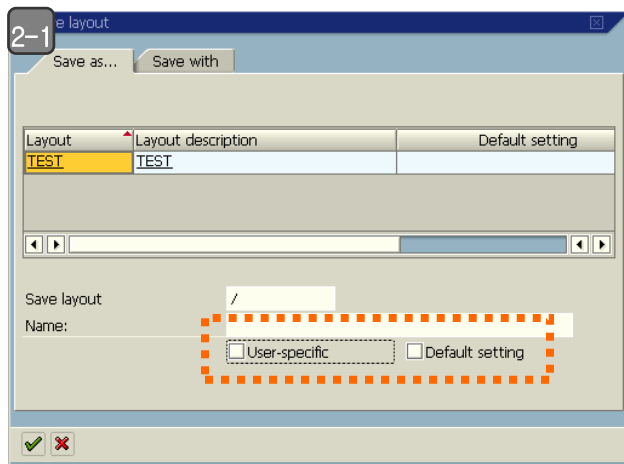


## 1. LAYOUT 저장

Save Layout 메뉴를 클릭하자.

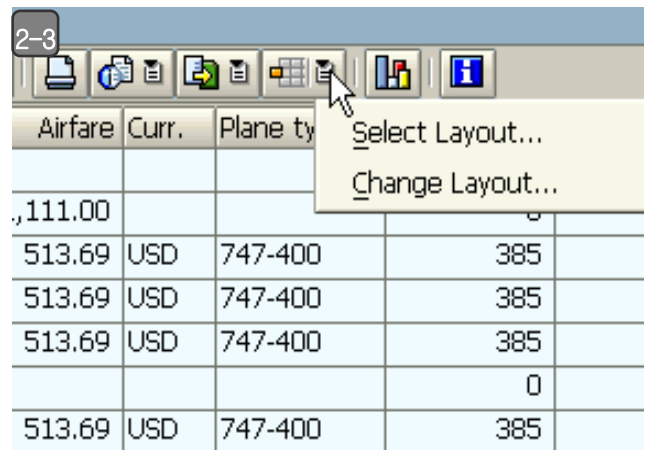
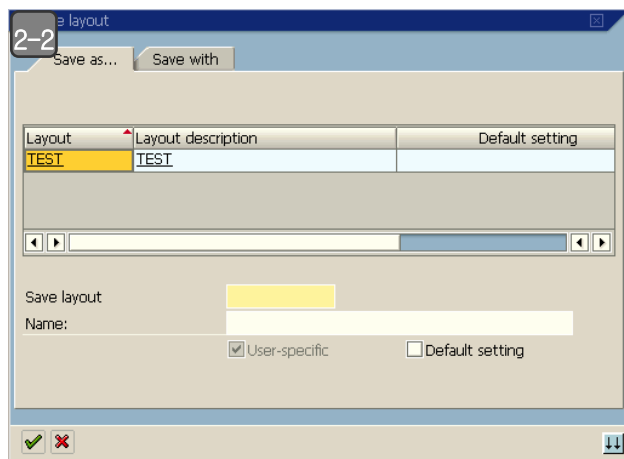
## 2. Save Layout

2-1. Z18\_003 예제에서 i\_save = 'A' 옵션을 준 것은 ALV LAYOUT을 변경하여 유저별로 사용하게 할 것이니 프로그램 기본 세팅으로 저장할것인지를 선택할수 있게 한다.



2-2. 는 i\_save = 'U' 본인의 ID의 Variant 설정만 가능하며, Default setting을 선택하더라도 본인의 layout 기본 세팅으로 설정된다.

2-3. 은 i\_save = ' ' 로 설정한것이며, layout을 변경할수는 있지만 저장이 불가능하다.





## ALV 레이아웃 선택

LAYOUT을 저장하였다면, 레포트 프로그램에서 레이아웃을 선택할수 있는 기능이 추가되어야 한다. LAYOUT을 참고하는 파라미터를 생성하여, AT SELECTION-SCREEN 이벤트에 Possible Entry를 띄워주는 소스를 추가해보자. [결과18-4-1]은 합계금액을 저장한 'TEST' 이름의 레이아웃을 보여주는 결과화면이다.

예제 18-4-1

```
REPORT z18_004 .

DATA : con1_ref TYPE REF TO
      cl_gui_custom_container.

DATA : grid1 TYPE REF TO cl_gui_alv_grid,
      gs_variant LIKE disvariant,
      gs_cs_variant LIKE disvariant,
      gt_sflight TYPE TABLE OF sflight.

PARAMETERS: p_vari LIKE disvariant-variant.

AT SELECTION-SCREEN ON VALUE-REQUEST FOR p_vari.

  GS_VARIANT-REPORT = SY-REPID.

  CALL FUNCTION 'REUSE_ALV_VARIANT_F4'
    EXPORTING
      is_variant = gs_variant
      i_save = 'A'
    IMPORTING
      es_variant = gs_cs_variant
  EXCEPTIONS
    not_found = 1
    program_error = 2
    OTHERS = 3.

  IF sy-subrc EQ 0.
    p_vari = gs_cs_variant-variant.
  ENDIF.
```

결과 18-4-1

### Program Z18\_004

| Layout | Layout description |
|--------|--------------------|
| /TEST  | TEST               |

### Program Z18\_004

| ID  | No.  | Flight Date | Airfare (Cur.) | Plane type | z    | Capacity | z   | Occupied | z | Booking total |
|---|------|-------------|----------------|------------|------|----------|-----|----------|---|---------------|
| SQ  | 988  | 2007.06.03  | 605.91         | USD        | A321 | 220      | 187 | 5        |   | 11,037.58     |
| UA  | 941  | 2006.03.21  | 1,068.00       | USD        | A321 | 220      | 187 |          |   | 205,803.60    |
| UA  | 941  | 2006.07.25  | 1,068.00       | USD        | A321 | 220      | 187 |          |   | 206,124.00    |
| UA  | 941  | 2006.09.28  | 1,068.00       | USD        | A321 | 220      | 188 |          |   | 207,726.00    |
| UA  | 941  | 2006.10.28  | 1,068.00       | USD        | A321 | 220      | 196 |          |   | 214,732.08    |
| UA  | 941  | 2006.11.07  | 1,068.00       | USD        | A321 | 220      | 195 |          |   | 214,411.68    |
| UA  | 941  | 2007.01.30  | 1,068.00       | USD        | A321 | 220      | 62  |          |   | 80,730.12     |
| UA  | 941  | 2007.04.03  | 1,068.00       | USD        | A321 | 220      | 39  |          |   | 57,797.44     |
| UA  | 941  | 2007.05.15  | 1,068.00       | USD        | A321 | 220      | 7   |          |   | 25,728.12     |
| UA  | 941  | 2007.06.05  | 1,068.00       | USD        | A321 | 220      | 10  |          |   | 7,817.76      |
| UA  | 3504 | 2006.03.17  | 1,068.00       | USD        | A321 | 220      | 180 |          |   | 198,487.80    |
| UA  | 3504 | 2006.07.21  | 1,068.00       | USD        | A321 | 220      | 185 |          |   | 202,225.80    |
| UA  | 3504 | 2006.09.28  | 1,068.00       | USD        | A321 | 220      | 187 |          |   | 206,017.20    |
| UA  | 3504 | 2006.10.28  | 1,068.00       | USD        | A321 | 220      | 182 |          |   | 200,997.60    |
| UA  | 3504 | 2006.11.03  | 1,068.00       | USD        | A321 | 220      | 185 |          |   | 203,304.48    |
| UA  | 3504 | 2007.01.26  | 1,068.00       | USD        | A321 | 220      | 15  |          |   | 33,545.88     |
| UA  | 3504 | 2007.03.30  | 1,068.00       | USD        | A321 | 220      | 18  |          |   | 0.00          |
| UA  | 3504 | 2007.05.11  | 1,068.00       | USD        | A321 | 220      | 9   |          |   | 27,041.76     |
| UA  | 3504 | 2007.06.01  | 1,068.00       | USD        | A321 | 220      | 18  |          |   | 0.00          |
| AUD 42,115 * 17,758 * 1,955,178.08<br>DEM 4,604,704.82<br>ITL 8,380,169.793<br>SGD 2,783,842.18<br>USD 4,171,764.72 |      |             |                |            |      |          |     |          |   |               |

### 4-1-3. I\_SAVE

이미 앞에서 설명을 했지만 간단히 값에 대한 정리만 하고 넘어가자.

| VALUE | 기능                      |
|-------|-------------------------|
| X     | 오직 Global 레이아웃 세팅만 가능함  |
| U     | 특정 유저에 한해서 레이아웃 세팅만 가능함 |
| A     | X 와 A 둘다 가능함            |
| SPACE | 레이아웃 저장을 하지 않음          |

### 4-1-4. I\_DEFAULT

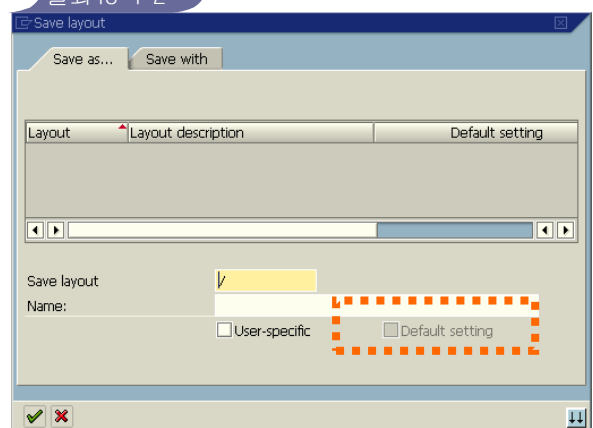
이 파라미터는 사용자가 Default Variant를 저장할수 있는지를 결정하게 한다.

| VALUE | 기능                        |
|-------|---------------------------|
| X     | Default variant 저장 가능함.   |
| SPACE | Default variant 저장이 불가능함. |

예제 18-4-2

```
REPORT z18_005 .
~~~~ 생략 ~~~~
CALL METHOD grid1->set_table_for_first_display
EXPORTING
    i_structure_name = 'SFLIGHT'
    i_save           = 'A'
    is_variant       = gs_variant
    i_default        = ''
CHANGING
    it_outtab        = gt_sflight.
```

결과 18-4-2



#### 4-1-5. IS\_LAYOUT

합계금액을 보여주는 등의 GRID Control의 화면 속성을 정의 한다. 레이아웃은 LVC\_S\_LAYO 타입의 구조체이다. SE11에서 조회해보자.

| Field name | Short description  | Purpose        |
|------------|--|----------------|
| CTAB_FNAME | 필드 CELL의 색상을 지정한다.   | Colors         |
| CWIDTH_OPT | 컬럼 길이를 지정한다.<br>Optimize column width  | GRID 속성        |
| DETAILINIT | 상세화면에서 기본값을 보여줄 것인지 결정한다.  | Interaction    |
| DETAILTITL | 상세화면의 Title bar를 보여준다.   | Interaction    |
| EXCP_CONDS | 예외 사항의 필드 sub Total을 보여준다.   | Exceptions     |
| EXCP_FNAME | 예외 사항 필드를 지정한다.  | Exceptions     |
| EXCP_LED   | 예외 사항 필드를 신호등 표시가 아니라, LED로 보여준다.  | Exceptions     |
| EXCP_ROLLN | 예외사항 필드에 대한 도움말을 표시한다.   | Exceptions     |
| GRID_TITLE | 타이틀 바의 내역을 지정한다.   | GRID 속성        |
| INFO_FNAME | ROW의 색상을 지정한다.   | Colors         |
| KEYHOT     | HOTSPOC으로 지정할 KEY 필드를 지정한다.  | Interaction    |
| NO_HEADERS | 컬럼 헤더를 보이지 않는다.  | GRID 속성        |
| NO_HGRIDLN | GRID의 수평선을 보이지 않는다.  | GRID 속성        |
| NO_MERGING | 컬럼을 소팅 할 경우 동일한 값의 CELL 합침을 막는다.   | GRID 속성        |
| NO_ROWMARK | GRID의 ROW를 선택할수 있는 버튼을 없앤다.<br>SEL_MODE = 'D' 이면 Row 버튼을 없앴<br>SEL_MODE = 'A' 이면 Column/Row 버튼을 없앴 | GRID 속성        |
| NO_TOOLBAR | 툴바를 보이지 않는다.   | GRID 속성        |
| NO_TOTLINE | Total 라인을 보이지 않는다.   | Totals Options |
| NO_VGRIDLN | Grid의 수직선을 보이지 않는다.  | GRID 속성        |
| NUMC_TOTAL | NUMC 필드의 합계금액을 보여준다.   | Totals Options |
| S_DRAGDROP | Drag & Drop 컨트롤을 세팅한다.( Row를 복사, 이동등의 기능)  | Interaction    |
| SEL_MODE   | Selection mode를 세팅한다.(A ,B ,C ,D, SPACE )  | GRID 속성        |
| SGL_CLK_HD | 컬럼 헤더를 클릭했을 경우 SORT를 수행한다.   | Interaction    |
| SMALLTITLE | Title size를 결정한다.  | GRID 속성        |
| TOTALS_BEH | 합계금액을 맨위의 ROW에 보여준다.   | Totals Options |
| ZEBRA      | ROW 단위별로 Stripped 패턴을 세팅한다.  | Colors         |

LAYOUT 속성을 몇가지만 설정해서 테스트 해보자. 이외의 파라미터는 직접 세팅하면서 실행해보면 어렵지 않게 이해할 수 있다. 먼저, lvc\_s\_layo 타입의 변수를 생성하자. Layout 속성을 변경하는 서브루틴(perform)을 추가하여 레이아웃 속성을 세팅한다. 그리고 ALV를 화면에 보여주는 메소드의 파라미터에 IS\_LAYOUT 을 추가하면 된다.

[결과18-4-3]의 위, 아래 그림을 비교해보면, 몇가지 설정이 변경된 것을 확인할 수 있다.

예제 18-4-3

```
REPORT z18_006 .
Data : gs_layout      TYPE lvc_s_layo.
~~
PERFORM setting_layout USING gs_layout.
~~
FORM setting_layout USING p_layout TYPE
lvc_s_layo.
*- General display options
p_layout-cwidth_opt = 'X'.
* TITLE BAR
p_layout-grid_title = 'LAYOUT TEST.'.
* Selection modes for SEL_MODE
p_layout-sel_mode = 'D'.
* Grid pattern
p_layout-zebra      = 'X'.
ENDFORM.                    " setting_layout
~
CALL METHOD grid1->set_table_for_first_display
EXPORTING
i_structure_name = 'SFLIGHT'
i_save           = 'A'
is_variant       = gs_variant
i_default        = ''
is_layout        = gs_layout
CHANGING
it_outtab        = gt_sflight.
~~
```

결과18-4-3

| ID | No. | Flight Date | Airfare   | Curr. | Plane type | Capacity | Occupied | Booking total | Capacity | Occupied | Capacity | Occupied |
|----|-----|-------------|-----------|-------|------------|----------|----------|---------------|----------|----------|----------|----------|
| AA | 17  | 2006.07.19  | 1,111.00  |       |            | 0        | 0        |               | 4        | 0        | 0        | 0        |
| AA | 17  | 2006.07.23  | 513.69    | USD   | 747-400    | 385      | 343      | 174,120.69    | 0        | 0        | 0        | 0        |
| AA | 17  | 2006.11.05  | 513.69    | USD   | 747-400    | 385      | 335      | 169,862.28    | 0        | 0        | 0        | 0        |
| AA | 17  | 2007.01.28  | 513.69    | USD   | 747-400    | 385      | 120      | 67,278.12     | 0        | 0        | 0        | 0        |
| AA | 17  | 2007.01.29  |           |       |            | 0        | 0        |               | 4        | 0        | 0        | 0        |
| AA | 17  | 2007.04.01  | 513.69    | USD   | 747-400    | 385      | 42       | 28,730.75     | 0        | 0        | 0        | 0        |
| AA | 17  | 2007.05.13  | 513.69    | USD   | 747-400    | 385      | 9        | 13,176.20     | 0        | 0        | 0        | 0        |
| AA | 17  | 2007.06.03  | 513.69    | USD   | 747-400    | 385      | 11       | 3,354.39      | 0        | 0        | 0        | 0        |
| AA | 64  | 2006.03.19  | 513.69    | USD   | 747-400    | 385      | 338      | 171,655.07    | 0        | 0        | 0        | 0        |
| AA | 64  | 2006.07.23  | 513.69    | USD   | 747-400    | 385      | 343      | 172,939.30    | 0        | 0        | 0        | 0        |
| AA | 64  | 2006.11.05  | 513.69    | USD   | 747-400    | 385      | 345      | 175,497.46    | 0        | 0        | 0        | 0        |
| AA | 64  | 2007.01.28  | 513.69    | USD   | 747-400    | 385      | 133      | 72,877.29     | 0        | 0        | 0        | 0        |
| AA | 64  | 2007.04.01  | 513.69    | USD   | 747-400    | 385      | 73       | 44,259.63     | 0        | 0        | 0        | 0        |
| AA | 64  | 2007.05.13  | 513.69    | USD   | 747-400    | 385      | 0        | 8,624.89      | 0        | 0        | 0        | 0        |
| AA | 64  | 2007.06.03  | 513.69    | USD   | 747-400    | 385      | 14       | 1,643.80      | 0        | 0        | 0        | 0        |
| AZ | 555 | 2006.03.18  | 360,202   | ITL   | A319       | 220      | 189      | 69,670,298    | 0        | 0        | 0        | 0        |
| AZ | 555 | 2006.07.22  | 360,202   | ITL   | A319       | 220      | 178      | 66,327,520    | 0        | 0        | 0        | 0        |
| AZ | 555 | 2006.11.04  | 360,202   | ITL   | A319       | 220      | 191      | 70,498,762    | 0        | 0        | 0        | 0        |
| AZ | 555 | 2007.01.27  | 360,202   | ITL   | A319       | 220      | 88       | 36,103,058    | 0        | 0        | 0        | 0        |
| AZ | 555 | 2007.03.31  | 360,202   | ITL   | A319       | 220      | 45       | 21,637,340    | 0        | 0        | 0        | 0        |
| AZ | 555 | 2007.05.12  | 360,202   | ITL   | A319       | 220      | 10       | 9,581,376     | 0        | 0        | 0        | 0        |
| AZ | 788 | 2006.03.19  | 2,667,445 | ITL   | 747-400    | 385      | 339      | 899,222,415   | 0        | 0        | 0        | 0        |

| ID | No. | Flight Date | Airfare   | Curr. | Plane type | Capacity | Occupied | Booking total |
|----|-----|-------------|-----------|-------|------------|----------|----------|---------------|
| AA | 17  | 2006.07.19  | 1,111.00  |       |            | 0        | 0        |               |
| AA | 17  | 2006.07.23  | 513.69    | USD   | 747-400    | 385      | 343      | 174,120.69    |
| AA | 17  | 2006.11.05  | 513.69    | USD   | 747-400    | 385      | 335      | 169,862.28    |
| AA | 17  | 2007.01.28  | 513.69    | USD   | 747-400    | 385      | 120      | 67,278.12     |
| AA | 17  | 2007.01.29  |           |       |            | 0        | 0        |               |
| AA | 17  | 2007.04.01  | 513.69    | USD   | 747-400    | 385      | 42       | 28,730.75     |
| AA | 17  | 2007.05.13  | 513.69    | USD   | 747-400    | 385      | 9        | 13,176.20     |
| AA | 17  | 2007.06.03  | 513.69    | USD   | 747-400    | 385      | 11       | 3,354.39      |
| AA | 64  | 2006.03.19  | 513.69    | USD   | 747-400    | 385      | 338      | 171,655.07    |
| AA | 64  | 2006.07.23  | 513.69    | USD   | 747-400    | 385      | 343      | 172,939.30    |
| AA | 64  | 2006.11.05  | 513.69    | USD   | 747-400    | 385      | 345      | 175,497.46    |
| AA | 64  | 2007.01.28  | 513.69    | USD   | 747-400    | 385      | 133      | 72,877.29     |
| AA | 64  | 2007.04.01  | 513.69    | USD   | 747-400    | 385      | 73       | 44,259.63     |
| AA | 64  | 2007.05.13  | 513.69    | USD   | 747-400    | 385      | 0        | 8,624.89      |
| AA | 64  | 2007.06.03  | 513.69    | USD   | 747-400    | 385      | 14       | 1,643.80      |
| AZ | 555 | 2006.03.18  | 360,202   | ITL   | A319       | 220      | 189      | 69,670,298    |
| AZ | 555 | 2006.07.22  | 360,202   | ITL   | A319       | 220      | 178      | 66,327,520    |
| AZ | 555 | 2006.11.04  | 360,202   | ITL   | A319       | 220      | 191      | 70,498,762    |
| AZ | 555 | 2007.01.27  | 360,202   | ITL   | A319       | 220      | 88       | 36,103,058    |
| AZ | 555 | 2007.03.31  | 360,202   | ITL   | A319       | 220      | 45       | 21,637,340    |
| AZ | 555 | 2007.05.12  | 360,202   | ITL   | A319       | 220      | 10       | 9,581,376     |
| AZ | 555 | 2007.06.02  | 360,202   | ITL   | A319       | 220      | 18       | 0             |
| AZ | 788 | 2006.03.19  | 2,667,445 | ITL   | 747-400    | 385      | 339      | 899,222,415   |

#### 4-1-6. IT\_OUTTAB

조회될 데이터의 Output 테이블을 정의한다. 다시말해서 ALV에 조회될 데이터를 가지고 있는 인터널 테이블을 지정하는 파라미터이다. 이미 앞에서 실습하였으므로 넘어가자.

#### 4-1-7. IT\_FIELDCATALOG

조회될 데이터의 타입 및 Output 테이블의 구조를 결정한다. 다루어야 할 내용이 많으므로 뒤에서 자세하게 설명하자.

#### 4-1-8. IT\_TOOLBAR\_EXCLUDING

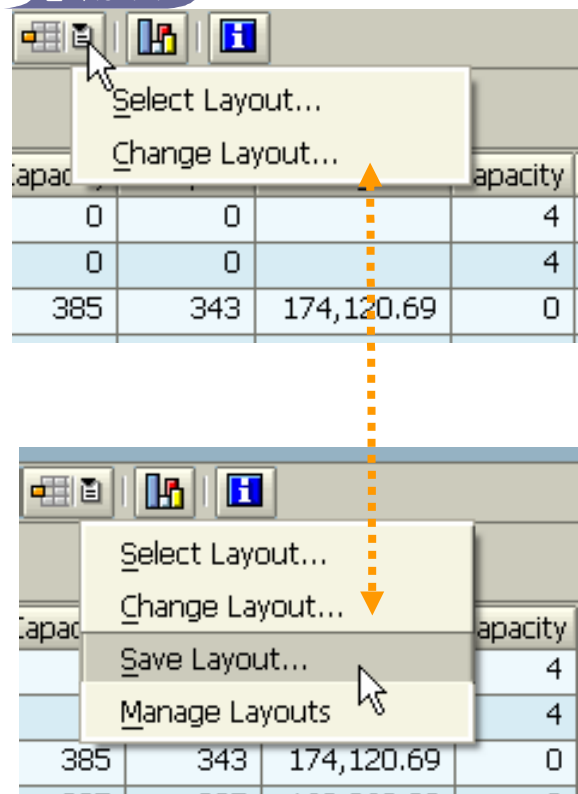
ALV Grid 컨트롤에서 숨기고 싶은 버튼이 있을 경우 사용한다. 예를들어 기능직 사원에게는 레이아웃을 저장할수 있는 버튼을 숨겨야 될 필요성이 발생할수 있다. 이럴 경우는 레이아웃 버튼을 사용함에 따라 보이고/숨기게 할수 있다.

ui\_functions 타입의 인터널 테이블을 선언하여 이 테이블에 Function 코드를 삽입하면 해당 코드는 삭제된다. 실습을 통해 이해를 돕자.

예제 18-4-4

```
REPORT z18_007 .
Data : gs_toolbar TYPE ui_functions.
~~
PERFORM setting_toolbar.
FORM setting_toolbar.
DATA: l_exclude TYPE ui_func.
l_exclude =
cl_gui_alv_grid=>mc_fc_save_variant.
APPEND l_exclude TO gs_toolbar.
l_exclude =
cl_gui_alv_grid=>mc_fc_maintain_variant.
APPEND l_exclude TO gs_toolbar.
ENDFORM. " setting_toolbar
~~
CALL METHOD grid1->set_table_for_first_display
EXPORTING
i_structure_name = 'SFLIGHT'
i_save           = 'A'
is_variant       = gs_variant
i_default        = ''
is_layout        = gs_layout
it_toolbar_excluding = gs_toolbar
CHANGING
it_outtab        = gt_sflight.
```

결과 18-4-4



실행 결과화면에서 Save Layout, Mange Layout 버튼이 사라졌다. 툴바 제거 기능에 대한 소스는 최대한 간단하고 쉽게 설명하기 위한 것이므로 다른 툴바들을 선택하여 직접 테스트 해보기 바란다.

[예제 18-4-4]에 사용된 TOOLBAR 버튼은 CL\_GUI\_ALV\_GRID의 속성 탭에 정의되어 있다. SE24에서 조회해보자. Associated Type 정의부분에서 UI\_FUNC로 정의된 속성들이 툴바에 관련된 속성이므로 필요 없는 툴바들은 소스를 추가해주면 된다.

**Class Builder: Display Class CL\_GUI\_ALV\_GRID**

Class interface: CL\_GUI\_ALV\_GRID    Implemented / Active

Properties   Interfaces   Friends   Attributes   Methods   Events   Internal types   Aliases

Filter

| Attribute             | Level   | Visi   | Mo... | Rea... | Typing | Associated Type | Description               | Initial value |
|-----------------------|---------|--------|-------|--------|--------|-----------------|---------------------------|---------------|
| MC_FC_MAINTAIN_VARIAN | Constan | Public |       |        | Type   | UI_FUNC         | Maintain Variants         | '&MAINTAIN'   |
| MC_FC_MAXIMUM         | Constan | Public |       |        | Type   | UI_FUNC         | Maximum                   | '&MAXIMUM'    |
| MC_FC_MINIMUM         | Constan | Public |       |        | Type   | UI_FUNC         | Minimum                   | '&MINIMUM'    |
| MC_FC_PC_FILE         | Constan | Public |       |        | Type   | UI_FUNC         | Export Local File         | '&PC'         |
| MC_FC_PRINT           | Constan | Public |       |        | Type   | UI_FUNC         | Print                     | '&PRINT'      |
| MC_FC_PRINT_BACK      | Constan | Public |       |        | Type   | UI_FUNC         | Print Backend             | '&PRINT_BAC   |
| MC_FC_PRINT_PREV      | Constan | Public |       |        | Type   | UI_FUNC         | Print Preview             | '&PRINT_BAC   |
| MC_FC_REFRESH         | Constan | Public |       |        | Type   | UI_FUNC         | Refresh                   | '&REFRESH'    |
| MC_FC_REPREP          | Constan | Public |       |        | Type   | UI_FUNC         | Report/Report Interface   | '&REPREP'     |
| MC_FC_SAVE_VARIANT    | Constan | Public |       |        | Type   | UI_FUNC         | Save Variant              | '&SAVE'       |
| MC_FC_SELECT_ALL      | Constan | Public |       |        | Type   | UI_FUNC         | Select All Rows           | '&ALL'        |
| MC_FC_SEND            | Constan | Public |       |        | Type   | UI_FUNC         | Send                      | '&SEND'       |
| MC_FC_SEPARATOR       | Constan | Public |       |        | Type   | UI_FUNC         | Separator                 | '&&SEP'       |
| MC_FC_SORT            | Constan | Public |       |        | Type   | UI_FUNC         | Sort                      | '&SORT'       |
| MC_FC_SORT_ASC        | Constan | Public |       |        | Type   | UI_FUNC         | Sort in Ascending Order   | '&SORT_ASC'   |
| MC_FC_SORT_DSC        | Constan | Public |       |        | Type   | UI_FUNC         | Sort in Descending Order  | '&SORT_DSC'   |
| MC_FC_SUBTOT          | Constan | Public |       |        | Type   | UI_FUNC         | Subtotals                 | '&SUBTOT'     |
| MC_FC_SUM             | Constan | Public |       |        | Type   | UI_FUNC         | Total                     | '&SUMC'       |
| MC_FC_TO_OFFICE       | Constan | Public |       |        | Type   | UI_FUNC         | Export Office             | '&ML'         |
| MC_FC_TO_REP_TREE     | Constan | Public |       |        | Type   | UI_FUNC         | Export Reporting Tree     | '&SERP'       |
| MC_FC_UNFIX_COLUMNS   | Constan | Public |       |        | Type   | UI_FUNC         | Unfreeze Columns          | '&CDF'        |
| MC_FC_URL_COPY_TO_CLI | Constan | Public |       |        | Type   | UI_FUNC         | Generate URL for RFC Call | '&URL_COPY'   |
| MC_FC_VARIANT_ADMIN   | Constan | Public |       |        | Type   | UI_FUNC         | Function Code             | '&VARI_ADMI   |
| MC_FC_VIEWS           | Constan | Public |       |        | Type   | UI_FUNC         | View Change               | '&VIEW'       |
| MC_FC_VIEW_CRYSTAL    | Constan | Public |       |        | Type   | UI_FUNC         | Crystal Preview Inplace   | '&VCRYSTAL'   |
| MC_FC_VIEW_EXCEL      | Constan | Public |       |        | Type   | UI_FUNC         | Excel Inplace             | '&VEXCEL'     |
| MC_FC_VIEW_GRID       | Constan | Public |       |        | Type   | UI_FUNC         | Grid Control              | '&VGRID'      |
| MC_FC_VIEW_LOTUS      | Constan | Public |       |        | Type   | UI_FUNC         | Lotus Inplace             | '&VLOTUS'     |
| MC_FC_WORD_PROCESSOR  | Constan | Public |       |        | Type   | UI_FUNC         | Word Processing           | '&AQW'        |
| MC_LYSTYLE_DRAG_DROP  | Constan | Public |       |        | Type   | RAW4            | Row Drag and Drop Enable  | '00000001'    |

▲ 그림 18-4-2. ALV TOLLBAR 버튼 조회

#### 4-1-9. IT\_SORT

ALV실행시 데이터가 정렬이 된 상태로 조회되도록 설정한다. LVC\_T\_SORT 타입으로 선언된 인터널 테이블을 선언하여 이 테이블에 정렬하고자하는 필드를 추가하면 된다. LVC\_T\_SORT 에서 “T”는 테이블을 의미하며 LVC\_S\_SORT 에서 “S”는 Structure를 의미한다. 소팅 테이블의 옵션중 subtot은 정렬필드기준으로 합계금액을 보여주고 전체 합계금액을 보여주는 것을 세팅한다. 실습을 통해 이해를 돕자.

예제 18-4-5

```
REPORT z18_008 .
Data :   gt_sort      TYPE lvc_t_sort.
~~
PERFORM setting_sort.
FORM setting_sort .
DATA : ls_sort TYPE lvc_s_sort.
ls_sort-spos = '1'.
ls_sort-fieldname = 'CARRID'.
ls_sort-up = 'X'.
ls_sort-subtot = 'X'.
APPEND ls_sort TO gt_sort.
ENDFORM.
~~
CALL METHOD grid1->
    set_table_for_first_display
EXPORTING
    i_structure_name      = 'SFLIGHT'
    i_save                = 'A'
    is_variant            = gs_variant
    i_default             = ''
    is_layout             = gs_layout
    it_toolbar_excluding = gs_toolbar
CHANGING
    it_outtab            = gt_sflight
    it_sort              = gt_sort.
~~
```

결과 18-4-5

| ID | No. | Flight Date | Airfare  | Curr. | Plane type | Σ | Capacity | Occupied |
|----|-----|-------------|----------|-------|------------|---|----------|----------|
| AA | 17  | 2006.03.19  |          |       |            |   | 0        | 0        |
|    | 17  | 2006.07.19  | 1,111.00 |       |            |   | 0        | 0        |
|    | 17  | 2006.07.23  | 513.69   | USD   | 747-400    |   | 385      | 343      |
|    | 17  | 2006.11.05  | 513.69   | USD   | 747-400    |   | 385      | 335      |
|    | 17  | 2007.01.28  | 513.69   | USD   | 747-400    |   | 385      | 120      |
|    | 17  | 2007.01.29  |          |       |            |   | 0        | 0        |
|    | 17  | 2007.04.01  | 513.69   | USD   | 747-400    |   | 385      | 42       |
|    | 17  | 2007.05.13  | 513.69   | USD   | 747-400    |   | 385      | 9        |
|    | 17  | 2007.06.03  | 513.69   | USD   | 747-400    |   | 385      | 11-      |
|    | 64  | 2006.03.19  | 513.69   | USD   | 747-400    |   | 385      | 338      |
|    | 64  | 2006.07.23  | 513.69   | USD   | 747-400    |   | 385      | 343      |
|    | 64  | 2006.11.05  | 513.69   | USD   | 747-400    |   | 385      | 345      |
|    | 64  | 2007.01.28  | 513.69   | USD   | 747-400    |   | 385      | 133      |
|    | 64  | 2007.04.01  | 513.69   | USD   | 747-400    |   | 385      | 73       |
|    | 64  | 2007.05.13  | 513.69   | USD   | 747-400    |   | 385      | 0        |
|    | 64  | 2007.06.03  | 513.69   | USD   | 747-400    |   | 385      | 14-      |
| AA |     |             |          |       |            |   | 5,005    |          |
| AZ | 555 | 2006.03.18  | 360,202  | ITL   | A319       |   | 220      | 189      |
|    | 555 | 2006.07.22  | 360,202  | ITL   | A319       |   | 220      | 178      |
|    | 555 | 2006.11.04  | 360,202  | ITL   | A319       |   | 220      | 191      |
|    | 555 | 2007.01.27  | 360,202  | ITL   | A319       |   | 220      | 88       |
|    | 555 | 2007.03.31  | 360,202  | ITL   | A319       |   | 220      | 45       |
|    | 555 | 2007.05.12  | 360,202  | ITL   | A319       |   | 220      | 10       |

| Component  | RTY | Component type | Data Type | Length | Decim | Short Description                                 |
|------------|-----|----------------|-----------|--------|-------|---|
| INCLUDE    |     | ALV_S_SORT     | NUMC      | 0      |       | Sort Criteria (for LVC and for KKBLO)             |
| SPOS       |     | SLIS_SPOS      | NUMC      | 2      |       | Sort sequence                                     |
| FIELDNAME  |     | LVC_FIELDNAME  | CHAR      | 30     |       | ALV control: Field name of internal table field   |
| UP         |     | CHAR1          | CHAR      | 1      |       | Single-character flag                             |
| DOWN       |     | CHAR1          | CHAR      | 1      |       | Single-character flag                             |
| GROUP      |     | SLIS_CTRL5     | CHAR      | 2      |       | Control Break: Insert Page Break, Underlines      |
| SUBTOT     |     | SLIS_DOSUB     | CHAR      | 1      |       | Output subtotal                                   |
| COMP       |     | CHAR1          | CHAR      | 1      |       | Single-character flag                             |
| EXPA       |     | CHAR1          | CHAR      | 1      |       | Single-character flag                             |
| SELTEXT    |     | SLIS_SCRIT     | CHAR      | 40     |       | Sort criterion                                    |
| OBLIGATORY |     | CHAR1          | CHAR      | 1      |       | Single-character flag                             |
| LEVEL      |     | INT4           | INT4      | 10     |       | Natural number                                    |
| NO_OUT     |     | CHAR1          | CHAR      | 1      |       | Single-character flag                             |
| INTOPT     |     | LVCIFLAG       | RAW       | 2      |       | ALV control: Internal optimization (INTERNAL USE) |

▲ 그림 18-4-3. 소팅 테이블의 속성



## 4-2. get\_current\_cell

ALV Grid Control에 커서가 놓인 위치의 값과 속성을 반환한다. 선택된 cell이 존재하지 않으면 row값은 0을 반환한다. ALV Grid 컨트롤은 두개의 row와 column의 인덱스를 반환한다. 하나는 현재 선택된 cell의 row와 column이고 다른 하나는 OUTPUT TABLE(인터널 테이블) row와 column의 인덱스이다. 이것은 필터링을 설정하거나 숨기기를 하였을 경우 실제 화면에 보이는 값과 인터널 테이블의 순서가 다를수 있기 때문이다.

CALL METHOD <ref.var. to CL\_GUI\_ALV\_GRID>->get\_current\_cell

```
IMPORTING
    E_ROW      = <var. of type I >
    E_VALUE    = <var. of type C >
    E_COL      = <var. of type I >
    ES_ROW_ID  = <structure of type LVC_S_ROW >
    ES_COL_ID  = <structure of type LVC_S_COL >.
```

| Parameter | Meaning                                 |
|-----------|---|
| E_ROW     | ALV Grid 컨트롤의 현재 ROW 인덱스                |
| E_VALUE   | ALV Grid 컨트롤의 현재 cell의 값                |
| E_COL     | ALV Grid 컨트롤의 현재 Column 인덱스             |
| ES_ROW_ID | Output Table의 현재 row type과 인덱스에 대한 정보구조 |
| ES_COL_ID | Output Table의 현재 Column과 필드명에 대한 정보 구조  |

### 예제 18-4-5

```
DATA: l_row    TYPE i,
      l_value  TYPE c,
      l_col    TYPE i,
      ls_row   TYPE lvc_s_row,
      ls_col   TYPE lvc_s_col,
      ls_roid  TYPE lvc_s_roid.
```

```
CALL METHOD grid1->get_current_cell
IMPORTING
    e_row      = l_row
    e_value    = l_value
    e_col      = l_col
    es_row_id  = ls_row
    es_col_id  = ls_col
    es_row_no  = ls_roid.
```

---

### 4-3. get\_frontend\_layout

현재 설정되어 있는 ALV Grid의 LAYOUT 정보를 가져온다.

```
CALL METHOD <ref.var. to CL_GUI_ALV_GRID > ->get_frontend_layout
```

```
IMPORTING  
    ES_LAYOUT = <structure of type LVC_S_LAYO > .
```

### 4-4. get\_selected\_cells

현재 선택된 cell들의 정보를 LVC\_T\_CELL 타입의 테이블로 반환한다. 현재 선택된 cell들의 필드명, index 등의 정보를 가져온다.

```
CALL METHOD <ref.var. to CL_GUI_ALV_GRID > ->get_selected_cells
```

```
IMPORTING  
    ET_CELL = <internal table of type LVC_T_CELL > .
```

### 4-5. get\_selected\_columns

선택된 컬럼들의 정보를 LVC\_T\_COL 타입의 테이블로 반환한다.

```
CALL METHOD <ref.var. to CL_GUI_ALV_GRID > ->get_selected_columns
```

```
IMPORTING  
    ET_INDEX_COLUMNS = <internal table of type LVC_T_COL > .
```

### 4-6. get\_selected\_rows

선택된 Row들의 정보를 LVC\_T\_ROW 타입의 테이블로 반환한다.

```
CALL METHOD <ref.var. to CL_GUI_ALV_GRID > ->get_selected_rows
```

```
IMPORTING  
    ET_INDEX_ROWS = <internal table of type LVC_T_ROW > .
```

## 4-7. refresh\_table\_display

ALV의 OUTPUT TABLE을 다시 보여줄때 사용되는 메소드이다. 데이터가 변경되거나 다시 SELECT 구문을 수행한 경우등에서 ALV 오브젝트를 생성하지 않고 데이터만 다시 보여주는 기능이다.

```
CALL METHOD <ref.var. to CL_GUI_ALV_GRID>->refresh_table_display
```

EXPORTING

```
IS_STABLE      = <structure of type LVC_S_STBL >
I_SOFT_REFRESH = <variable of type CHAR01 >.
```

| Parameter      | Meaning  |
|----------------|--|
| IS_STABLE      | ROW와 컬럼 위치를 기억하여 재조회한후 이전의 위치에 화면을 보이게 한다.                       |
| I_SOFT_REFRESH | Sort, Filter, Sum등 현 ALV Grid의 레이아웃 세팅을 그대로 유지하면서 refresh를 실행한다. |

### 예제 18-4-6

```
DATA : ls_scroll TYPE lvc_s_stbl.
      ls_scroll-row = 'X'.
      ls_scroll-col = 'X'.
```

```
CALL METHOD grid1->refresh_table_display
EXPORTING
  i_soft_refresh = 'X'
  is_stable      = ls_scroll.
```

## 4-8. set\_frontend\_layout

ALV Grid 레이아웃을 변경한다. 이 메소드를 호출한후에 REFRESH\_TABLE\_DISPLAY 메소드를 호출하여야 한다.

```
CALL METHOD <ref.var. to CL_GUI_ALV_GRID > ->set_frontend_layout
```

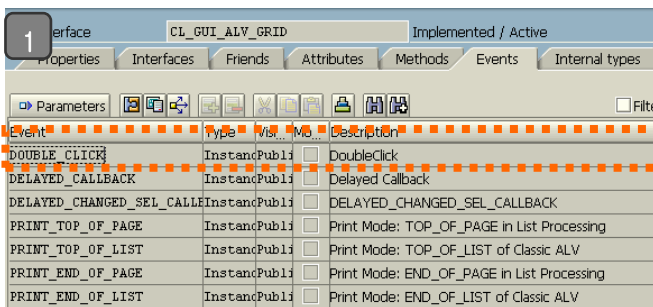
EXPORTING

```
IS_LAYOUT = <structure of type LVC_S_LAYOUT > .
```

이 외에도 소스레벨에서 Function을 수행할수 있는 set\_user\_command 메소드, 소팅정보를 가져오고 세팅 하는 get\_sort\_criteria와 set\_sort\_criteria, Variant 정보를 가져오는 get\_variant 메소드등 이 존재한다.

## 5. ALV 이벤트

ALV Grid에서 HOSTPOT, 더블클릭등의 유저 액션에 반응하는 이벤트를 추가할수 있다. 17장에서 이벤트 개념에 대해서 학습하였듯이, 클래스간에 이벤트를 등록하기 위해서는 **이벤트를 선언**하고, **이벤트 핸들러 메소드**를 정의하고, **이벤트 핸들러 메소드를 등록**해야 한다. 이해를 돕기 위해 순서대로 간략히 설명한후에 실습 예제를 학습하자.



### 1. 이벤트 선언

클래스빌더에서 CL\_GUI\_ALV\_GRID를 조회하면 DOUBLE\_CLICK 이라는 이벤트가 선언되어 있다.

### 2. 이벤트 핸들러 메소드 정의

프로그램내에서 이벤트가 발생하였을시 반응는 이벤트 핸들러 메소드를 정의한다.

```

2 class lcl_event_receiver definition.
  public section.
    methods:
      handle_double_click
        for event double_click
        of cl_gui_alv_grid
          importing e_row e_column.
endclass.
  
```

### 3. 이벤트 핸들러 메소드 등록(Register)

이벤트를 실행하기 위해 ALV Grid에 이벤트 핸들러 메소드를 등록한다.

### 4. 이벤트 호출

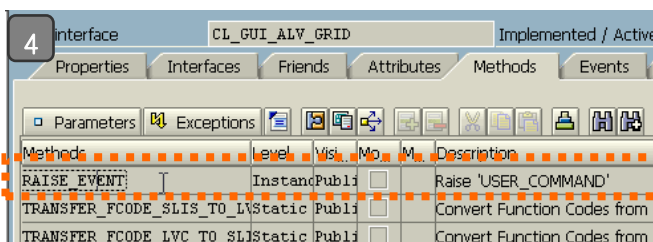
CL\_GUI\_ALV\_GRID의 화면을 더블클릭하게 되면 RAISE\_EVENT라는 메소드가 실행되고 이 메소드는 USER COMMAND에 해당하는 이벤트를 발생시키게 된다.

```

3 DATA :event_receiver type ref to
                                     lcl_event_receiver.
create object event_receiver.
set handler
  event_receiver->handle_double_click for grid1.
  
```

### 5. 이벤트 핸들러 메소드 실행

이벤트가 호출되어 이벤트 핸들러 메소드가 실행된다. class lcl\_event\_receiver implemnt에서 메소드의 행위를 정의하게 되면 ALV에서 더블클릭에 작동하는 소스 스크립트를 완성하게 된다.



method raise\_event .

```

  raise event after_user_command exporting
    e_ucomm = i_ucomm
  e_not_processed = i_not_processed.endmethod.
endmethod.
  
```

## 5-1. double\_click 이벤트

ALV 화면을 조회하여 CELL을 더블클릭 할 경우 화면을 빠져 나오는 이벤트 예제를 실습해보자.  
이미 앞에서 개념과 절차에 대해서 설명했으므로 아주 간단하게 작업할수 있다.

| Parameter      | Meaning  |
|----------------|--|
| IS_STABLE      | ROW와 컬럼 위치를 기억하여 재조회한후 이전의 위치에 화면을 보이게 한다.                       |
| I_SOFT_REFRESH | Sort, Filter, Sum등 현 ALV Grid의 레이아웃 세팅을 그대로 유지하면서 refresh를 실행한다. |

### 예제 18-5-1

```

1  REPORT z18_011 .
   CLASS lcl_event_receiver DEFINITION.
     PUBLIC SECTION.
       METHODS:
         handle_double_click
           FOR EVENT double_click OF cl_gui_alv_grid
             IMPORTING e_row e_column.
   ENDCLASS.

   CLASS lcl_event_receiver IMPLEMENTATION.
     METHOD handle_double_click.
       LEAVE TO SCREEN 0.
     ENDMETHOD.
   ENDCLASS.

2  DATA : event_receiver type ref to lcl_event_receiver.

```

1. 이벤트 핸들러 메소드를 포함하는 클래스를 정의하고 IMPLEMENT 한다.  
이벤트 핸들러 메소드를 선언하고 기술한다.

2. 클래스를 참고하는 객체 참고 변수를 정의한다.

```

3  create object event_receiver.
     set handler event_receiver->handle_double_click for
       grid1.

```

3. 오브젝트를 생성하여 이벤트 핸들러 메소드를 등록하자.  
그리고 프로그램을 실행하여 ALV Grid 의 임의의 셀을 더블클릭하면 프로그램을 빠져 나가게 된다.

## 5-2. hotspot\_click 이벤트

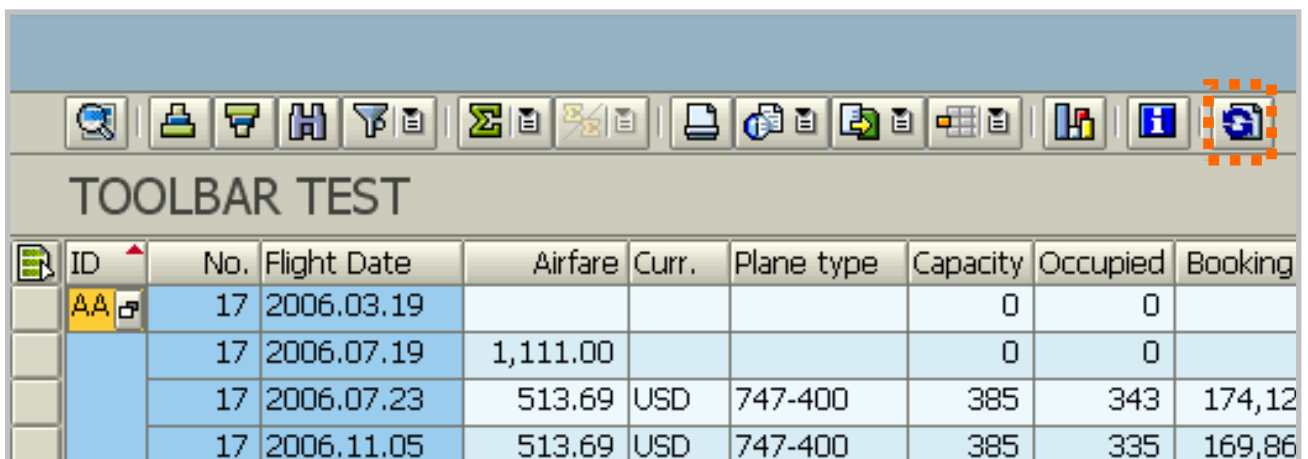
Hotspot으로 선언된 컬럼을 마우스-클릭할 경우에 반응하는 이벤트이다. 해당 컬럼은 필드카타로그 선언시 HOTSPOT 속성으로 선언되어야 한다. 필드카타로그는 뒷부분에서 자세하게 다룬다.

| Parameter                            | Meaning                    |
|--------------------------------------|----------------------------|
| E_ROW_ID<br>TYPE REF TO LVC_S_ROW    | Structure with row index   |
| E_COLUMN_ID<br>TYPE REF TO LVC_S_COL | Structure with column name |

## 5-3. toolbar 이벤트

ALV가 기본적으로 제공하는 버튼 이외에 프로그래머가 추가적인 버튼 추가할수 있다. Toolbar 이벤트는 ALV Grid에 단순히 버튼만 추가하는 것이고 버튼을 클릭했을 경우의 동작은 user\_command 이벤트에서 수행하게 된다. [그림 18-5-1]에서 Refresh 버튼이 추가된 것을 예제를 통해 학습해보자.

| Parameter   | Meaning   |
|---|---|
| E_OBJECT<br>Type Ref To<br>CL_ALV_EVENT_TOOLBAR_SET | The object contains only one attribute with a table for the functions of the toolbar.   |
| E_INTERACTIVE<br>Type CHAR01                        | If this flag is set, you triggered the event using method set_toolbar_interactive . If this flag is not set, the event was triggered by the ALV grid control. |



▲ 그림 18-5-1. TOOLBAR 버튼추가

```

3  REPORT z18_012 .
1  TYPE-POOLS: icon.
   CLASS lcl_event_receiver DEFINITION.
     PUBLIC SECTION.
       METHODS : handle_toolbar
                 FOR EVENT toolbar OF cl_gui_alv_grid
                 IMPORTING e_object e_interactive.
   ENDCLASS.

2  CLASS lcl_event_receiver IMPLEMENTATION.
     DATA: ls_toolbar TYPE stb_button.

     CLEAR ls_toolbar.
     ls_toolbar-butn_type = 3.   “수직 구분자
     APPEND ls_toolbar TO e_object->mt_toolbar.

     CLEAR ls_toolbar.
     ls_toolbar-function = 'RESH'.
     ls_toolbar-icon      = icon_refresh.
     ls_toolbar-quickinfo = 'Refresh'.
     ls_toolbar-text      = ' '.
     ls_toolbar-disabled = ' '.
     APPEND ls_toolbar TO e_object->mt_toolbar..
   ENDCLASS.
   ~~

4  create object event_receiver.
     SET HANDLER event_receiver->handle_toolbar FOR grid1.

```

- 이벤트 핸들러 메소드를 포함하는 클래스를 정의하고 IMPLEMENT 한다. 이벤트 핸들러 메소드를 선언하고 기술한다.
- 화면에 REFRESH 버튼을 추가하는 소스를 추가한다. ICON에 ICON\_REFRESH 구문을 이용하기 위해서는 ICON TYPE-POOL을 선언하여야 한다.
- TYPE-POOLS ICON을 선언하여 아이콘의 시스템 ID를 쉽게 알아 볼수 있도록 해준다. 실제 ICON\_REFRESH의 시스템 ID는 '@42@' 이다.
- 오브젝트를 생성하여 이벤트 핸들러 메소드를 등록하자. 그리고 프로그램을 실행하면 [그림 18-5-1] 과 같이 화면에 버튼이 추가되었음을 확인할수 있다.



#### 5-4. user\_command 이벤트

Toolbar 이벤트에서 추가된 버튼에 기능을 추가하는 이벤트 이다. Refresh 버튼을 클릭하였을 경우 데이터를 새로 읽어오는 로직을 추가해보자.

관계된 이벤트로 after\_user\_command, before\_user\_command 가 존재한다.

| Parameter                | Meaning                                 |
|--------------------------|---|
| E_UCOMM<br>TYPE SY-UCOMM | Function code for self-defined function |

예제 18-5-3

```

REPORT z18_013 .
TYPE-POOLS: icon.
CLASS lcl_event_receiver DEFINITION.
  PUBLIC SECTION.

      METHODS : handle_command
                  FOR EVENT user_command OF cl_gui_alv_grid
                  IMPORTING e_ucomm.

ENDCLASS.

CLASS lcl_event_receiver IMPLEMENTATION.
  METHOD handle_command.
    CASE e_ucomm.
      WHEN 'RESH'.
        SELECT * FROM sflight INTO TABLE gt_sflight.
        l_scroll-row = 'X'.
        l_scroll-col = 'X'.
        CALL METHOD grid1->refresh_table_display
          EXPORTING
            i_soft_refresh = ''
            is_stable      = l_scroll.
      ENDCASE.
    ENDMETHOD.
  ENDCLASS.
  ~~
  create object event_receiver.
  SET HANDLER event_receiver->handle_command FOR grid1.
  
```

1. Refresh 버튼을 클릭하면 데이터를 다시 SELECT한다.(DB의 내용이 변경되었다면 새로운 내용을 조회)
2. ALV Grid 를 재조회 한다. 4장 메소드에서 이미 학습하였다.

## 5-5. onDrag, onDrop 이벤트

ALV Grid 내에서 Drag & Drop을 수행할때 작동하는 Drag 관련 이벤트이다.

| Parameter                                    | Meaning  |
|--|--|
| E_ROW<br>Type LVC_S_ROW                      | Structure with the index of the row dragged    |
| E_COLUMN<br>Type LVC_S_COL                   | Structure with the index of the column dragged |
| E_DRAGDROPBJ<br>Type Ref To CL_DRAGDROPBJECT | 유저액션(copy, move)정보등을 포함한다.                     |

▲ 그림 18-5-1. TOOLBAR 버튼추가

[그림 18-5-1]에서 ID가 AZ로 시작하는 한 ROW를 Drag & Drop 하여 ALV 리스트에 맨위로 올리는 프로그램이다. 소스코드를 설명하려면 많은 분량이 필요하므로 간단히 프로그램명만 공유하니 직접 학습하며 학습하기 바란다. 해당 코드는 한 ROW를 Drag & Drop을 실행하면 인터널 테이블에 추가하여 화면을 REFRESH 하는 것이다. 이외에도 다음과 같은

context\_menu\_request, delayed\_callback, delayed\_changed\_sel\_callback, onDropComplete, onDropGetFlavor, print\_end\_of\_list, print\_end\_of\_page, print\_top\_of\_list, print\_top\_of\_page, subtotal\_text 이벤트가 더 존재하지만 이벤트 사용법을 이해하였다면 적용하는데 큰 어려움이 없으므로 다음장으로 넘어가자.

### 예제 18-5-4

REPORT z18\_014 .

~~

CREATE OBJECT g\_application.

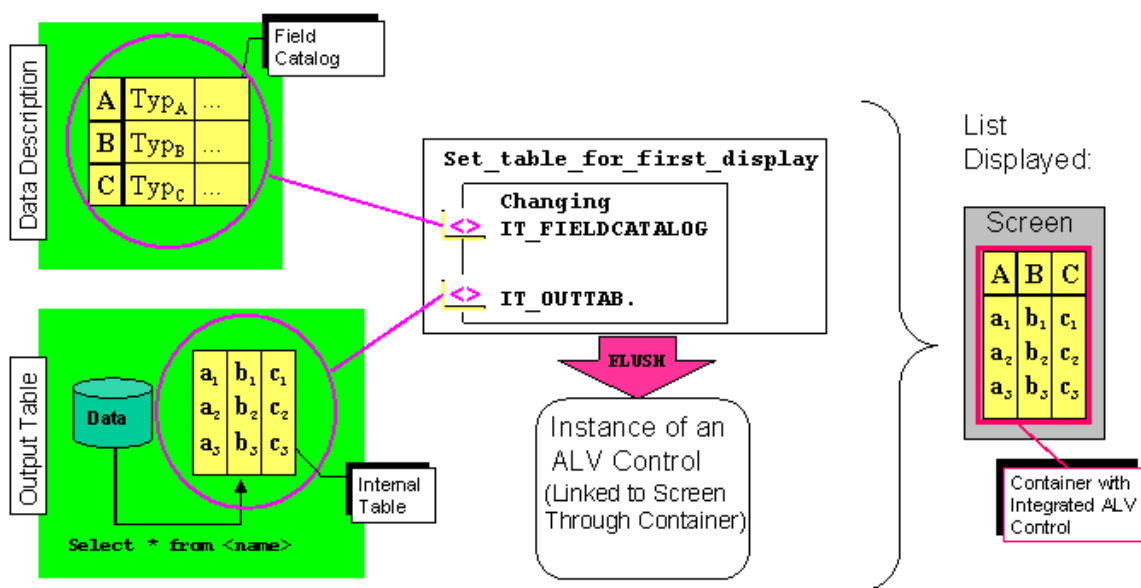
SET HANDLER g\_application->handle\_alv\_drag FOR grid1.

SET HANDLER g\_application->handle\_alv\_drop FOR grid1.

~~

## 6. 필드카타로그

필드카타로그는 ALV에서 조회되는 컬럼들의 필드정보를 포함하고 있는 **LVC\_T\_FCAT** 타입의 테이블이다. ALV는 필드카타로그 테이블을 이용해 필드 타입을 인식하게 된다. 예를들어, 필드가 number 타입인지 char 타입인지 구분하며, 화면에 보여줄 필드 길이를 지정하고, 체크박스-레디오 버튼으로 보이게 하고, 필드 수정이 가능하게 하는 등의 많은 역할을 하게 된다.



▲ 그림 18-6-1. TOOLBAR 버튼추가

[그림 18-6-1]에서 Data Description 부분은 필드카타로그를 잘 설명하고 있다. Output 테이블에서 select 한 결과를 필드카타로그의 필드 정보를 이용해 ALV 화면에 보여준다. [예제 18-2-2]에서는 STURCTR 구조를 이용해 OUTPUT 테이블의 필드를 구성하였으나 이번장에서는 개발자가 직접 필드카타로그를 구성하는 실습을 하게 된다.

```
Call method grid1->
set_table_for_first_display
EXPORTING
    I_STRUCTURE_NAME = 'SFLIGHT'
CHANGING
    IT_OUTTAB = gt_sflicht.
```

```
Call method grid1->
set_table_for_first_display
CHANGING
    IT_FIELDCATALOG = gt_catalog
    IT_OUTTAB = gt_sflicht.
```

---

필드카타로그를 정의하는 방법은 3가지가 존재한다.

- ABAP Data Dictionary 구조체를 이용하는 방법
- 프로그램 내에서 스크립트로(수동으로) 구성하는 방법
- 위 두가지 방법을 혼합하여 사용하는 방법

각각의 방법에 대해서 필드카타로그를 정의하는 방법에 대해서 알아보자.

## 6-1. ABAP Dictionary(구조체, 테이블)를 이용하는 방법

이미 앞의 예에서 테이블 Data Dictionary 구조체를 그대로 필드카타로그로 정의하는 방법은 학습하였다. ABAP Dictionary의 구조체와 테이블의 필드를 ALV OUTPUT 테이블에 그대로 보여주는 방법이다. 앞에서 살펴본 예제에서는 SFLIGHT 테이블 하나에서만 데이터를 읽어와 인터널 테이블에 데이터를 그대로 화면에 보여주는 프로그램이었다. 그러나 대부분의 실무 프로그램에서는 여러 개의 테이블에서 데이터를 가져오기때문에 테이블 구조를 필드카타로그를 그대로 사용할수 있는 경우가 그리 많지않다. 이러한 경우에는 STRUCTURE를 생성하여 인터널테이블을 이 구조체를 참고하여 생성하게 된다. 그러나 하나의 프로그램에서만 필요한 구조체라면 굳이 ABAP Dictionary 구조체를 생성할 필요는 없다. 이러한 경우에는(뒤에서 학습하게 되겠지만) 인터널 테이블 구조를 그대로 필드카타로그로 정의할수 있다.

```
CALL METHOD < ref.var. to CL_GUI_ALV_GRID>->set_table_for_first_display
  EXPORTING
    I_STRUCTURE_NAME      = < string of type DD02L-TABNAME>
    IS_VARIANT            = < structure of type DISVARIANT>
    I_SAVE                = < var. of type CHAR01>
    I_DEFAULT             = < var. of type CHAR01>
    IS_LAYOUT             = < structure of type LVC_S_LAYO>
    IS_PRINT              = < structure of type LVC_S_PRNT>
    IT_SPECIAL_GROUPS     = < internal table of type LVC_T_SGRP>
    IT_TOOLBAR_EXCLUDING = < internal table of type UI_FUNCTIONS>
  CHANGING
    IT_OUTTAB             = < internal table>
    IT_FIELDCATALOG       = < internal table of type LVC_T_FCAT>
    IT_FILTER             = < internal table of type LVC_T_FILT>
```

## 6-2. 필드카타로그를 수동으로 구성하는 방법

DATA Dictionary의 모든 필드를 필드카타로그로 보여주고 싶지 않은 경우가 있다. 이러한 경우에는 스크립트(수동으로)를 통해 필요한 필드만 카타로그로 정의할 수 있다. 그러나 각 필드의 이름, 타입 등을 스크립트로 모두 정의하여야 하므로 프로그램 수정(유지보수)에 많은 시간이 소요되는 단점이 있다.

| Output table fields <i>with</i> DDIC Reference | Output table fields <i>without</i> DDIC reference | Explanation           |
|--|---|-----------------------|
| FIELDNAME                                      | FIELDNAME   | OUTPUT 테이블의 필드이름      |
| REF_TABNAME                                    |   | 참고할 구조체의 DDIC 이름      |
| REF_FIELDNAME                                  |   | 참고할 구조체의 DDIC 필드이름    |
|  | INTTYPE   | OUTPUT 테이블의 DATA TYPE |
|  | OUTPUTLEN   | 컬럼 길이                 |
|  | COLTEXT   | 컬럼 헤더 텍스트             |
|  | SELTEXT   | Variant 조회시 컬럼 내역     |

```
CALL METHOD < ref.var. to CL_GUI_ALV_GRID>->set_table_for_first_display
EXPORTING
    IS_VARIANT          = < structure of type DISVARIANT>
    I_SAVE              = < var. of type CHAR01>
    I_DEFAULT           = < var. of type CHAR01>
    IS_LAYOUT           = < structure of type LVC_S_LAYO>
    IS_PRINT            = < structure of type LVC_S_PRNT>
    IT_SPECIAL_GROUPS   = < internal table of type LVC_T_SGRP>
    IT_TOOLBAR_EXCLUDING = < internal table of type UI_FUNCTIONS>
CHANGING
    IT_OUTTAB           = < internal table>
    IT_FIELDCATALOG     = < internal table of type LVC_T_FCAT>
    IT_SORT             = < internal table of type LVC_T_SORT>
    IT_FILTER           = < internal table of type LVC_T_FILT>
```

[예제 18-8-1]에서는 CARRID, CONNID, PRICE 필드만 카탈로그로 구성하여 화면에 보여지도록 구성하게 된다.

#### 예제 18-6-1

```
REPORT z18_017 .
Data :   gt_fieldcat  TYPE lvc_t_fcat.
~~
PERFORM setting_catalog.
form setting_catalog .
data ls_fieldcat TYPE lvc_s_fcat.

1 ls_fieldcat-fieldname = 'CARRID'.
ls_fieldcat-COLTEXT = 'Carrid ID'.
ls_fieldcat-JUST = 'L'.
ls_fieldcat-KEY = 'X'.
ls_fieldcat-OUTPUTLEN = '2'.
APPEND LS_FIELD CAT TO GT_FIELD CAT.

ls_fieldcat-fieldname = 'CONNID'.
ls_fieldcat-COLTEXT = 'Flight Number'.
ls_fieldcat-JUST = 'C'.
ls_fieldcat-KEY = 'X'.
ls_fieldcat-OUTPUTLEN = '4'.
APPEND LS_FIELD CAT TO GT_FIELD CAT.

ls_fieldcat-fieldname = 'PRICE'.
ls_fieldcat-COLTEXT = 'Airfare'.
ls_fieldcat-JUST = 'R'.
ls_fieldcat-KEY = ' '.
ls_fieldcat-OUTPUTLEN = '15'.
APPEND LS_FIELD CAT TO GT_FIELD CAT.
endform.
~~
```

#### CALL METHOD

```
2 grid1->set_table_for_first_display
EXPORTING
*   i_structure_name   = 'SFLIGHT'
i_save                 = 'A'
is_variant              = gs_variant
i_default               = ' '
is_layout               = gs_layout
it_toolbar_excluding    = gs_toolbar
CHANGING
it_outtab               = gt_sflight
IT_FIELD CATALOG        = gt_fieldcat
it_sort                 = gt_sort.
~~
```

#### 결과 18-6-1

| Carrid ID | Flight Number | Airfare   |
|-----------|---------------|-----------|
| AA        | 17            | 0.00      |
|           | 17            | 1,111.00  |
|           | 17            | 513.69    |
|           | 17            | 513.69    |
|           | 17            | 513.69    |
|           | 17            | 0.00      |
|           | 17            | 513.69    |
|           | 17            | 513.69    |
|           | 17            | 513.69    |
|           | 64            | 513.69    |
|           | 64            | 513.69    |
|           | 64            | 513.69    |
|           | 64            | 513.69    |
|           | 64            | 513.69    |
|           | 64            | 513.69    |
|           | 64            | 513.69    |
| AZ        | 555           | 3,602.02  |
|           | 555           | 3,602.02  |
|           | 555           | 3,602.02  |
|           | 555           | 3,602.02  |
|           | 555           | 3,602.02  |
|           | 555           | 3,602.02  |
|           | 555           | 3,602.02  |
|           | 555           | 3,602.02  |
| 78R       |               | 26,674.45 |

### 6-3. 구조체 이용과 필드카타로그를 동시에 사용

구조체와 필드카타로그 두가지를 혼합하여 사용할 수 있다. 구조체의 필드이외의 사용자 정의 필드가 더 필요한 경우에 사용하기 적합하다. 이때 구조체와 필드카타로그에 동일한 필드가 존재하게 되면, 필드카타로그에서 정의한 필드가 높은 우선순위를 가지게 된다. [예제 18-8-2]를 이용해 아시아나, 대한항공과 같은 항공회사를 화면에 보여주기 위해 필드를 추가해보자.

예제 18-6-2

```
REPORT z18_018 .
Data :   gt_fieldcat   TYPE lvc_t_fcat.
~~
PERFORM setting_catalog.
form setting_catalog .
    ls_fieldcat-fieldname = 'COMPANY'.
    ls_fieldcat-COLTEXT = 'COMPANY INFO'.
    ls_fieldcat-JUST = 'C'.
    ls_fieldcat-KEY = 'X'.
    ls_fieldcat-OUTPUTLEN = '6'.

    APPEND LS_FIELDCAT TO GT_FIELDCAT.
endform.
~~
CALL METHOD
    grid1->set_table_for_first_display
EXPORTING
    i_structure_name      = 'SFLIGHT'
    i_save                = 'A'
    is_variant            = gs_variant
    i_default              = ' '
    is_layout             = gs_layout
    it_toolbar_excluding  = gs_toolbar
CHANGING
    it_outtab             = gt_sflight
    IT_FIELDCATALOG       = gt_fieldcat
    it_sort               = gt_sort.
~~
```

결과 18-6-1

| No. | Flight Date | Airfare  | Curr. | Plane type | Capacity | Occupied | Booking total | Capacity | Occupied |
|-----|-------------|----------|-------|------------|----------|----------|---------------|----------|----------|
| 17  | 2006.03.19  |          |       |            | 0        | 0        |               | 4        | 0        |
| 17  | 2006.07.19  | 1,111.00 |       |            | 0        | 0        |               | 4        | 0        |
| 17  | 2006.07.23  | 513.69   | USD   | 747-400    | 365      | 343      | 174,120.69    | 0        | 0        |
| 17  | 2006.11.05  | 513.69   | USD   | 747-400    | 365      | 335      | 169,862.28    | 0        | 0        |
| 17  | 2007.01.28  | 513.69   | USD   | 747-400    | 365      | 120      | 67,278.12     | 0        | 0        |
| 17  | 2007.01.29  |          |       |            | 0        | 0        |               | 4        | 0        |
| 17  | 2007.04.01  | 513.69   | USD   | 747-400    | 365      | 42       | 28,730.75     | 0        | 0        |
| 17  | 2007.05.13  | 513.69   | USD   | 747-400    | 365      | 9        | 13,176.20     | 0        | 0        |
| 17  | 2007.06.03  | 513.69   | USD   | 747-400    | 365      | 11       | 3,354.39      | 0        | 0        |
| 64  | 2006.03.19  | 513.69   | USD   | 747-400    | 365      | 338      | 171,655.07    | 0        | 0        |
| 64  | 2006.07.23  | 513.69   | USD   | 747-400    | 365      | 343      | 172,939.30    | 0        | 0        |
| 64  | 2006.11.05  | 513.69   | USD   | 747-400    | 365      | 345      | 175,497.46    | 0        | 0        |
| 64  | 2007.01.28  | 513.69   | USD   | 747-400    | 365      | 133      | 72,877.29     | 0        | 0        |
| 64  | 2007.04.01  | 513.69   | USD   | 747-400    | 365      | 73       | 44,259.63     | 0        | 0        |
| 64  | 2007.05.13  | 513.69   | USD   | 747-400    | 365      | 0        | 8,624.89      | 0        | 0        |
| 64  | 2007.06.03  | 513.69   | USD   | 747-400    | 365      | 14       | 1,643.80      | 0        | 0        |
| 555 | 2006.03.18  | 360,202  | ITL   | A319       | 220      | 189      | 69,670,298    | 0        | 0        |
| 555 | 2006.07.22  | 360,202  | ITL   | A319       | 220      | 178      | 66,327,620    | 0        | 0        |
| 555 | 2006.11.04  | 360,202  | ITL   | A319       | 220      | 191      | 70,498,762    | 0        | 0        |
| 555 | 2007.01.27  | 360,202  | ITL   | A319       | 220      | 88       | 36,103,058    | 0        | 0        |
| 555 | 2007.03.31  | 360,202  | ITL   | A319       | 220      | 45       | 21,637,340    | 0        | 0        |
| 555 | 2007.05.12  | 360,202  | ITL   | A319       | 220      | 10       | 9,581,376     | 0        | 0        |
| 555 | 2007.06.20  | 360,202  | ITL   | A319       | 220      | 10       | 9,581,376     | 0        | 0        |

Compay 필드가 제일 먼저 나오는 것은 필드카타로그가 구조체보다 우선순위가 높기 때문이다. 그러므로 구조체에서 이미 선언된 필드를 변경하고 싶을 경우에는 필드 카타로그에서 다시 선언하여 변경해주면 된다. 예를들어 CARRID를 보여주고 싶지 않을 경우에는 다음 구문과 같이 사용하면 된다.

```
ls_fieldcat-fieldname = 'CARRID'.
ls_fieldcat-NO_OUT = 'X'.
APPEND LS_FIELDCAT TO GT_FIELDCAT.
```

이외의 필드카타로그 속성은 SE11에서 lvc\_s\_fcat를 조회하여 Description을 보면 어떠한 역할을 하는지 짐작할 수 있다. 필드 카타로그의 정의와 작동 원리를 이용하였다면 얼마든지 적용할 수 있을 것이다. 필드카타로그의 속성은 [표 18-6-1]과 같다.



| 카타로그 속성           | 내역                         | 사용목적               |
|-------------------|----------------------------|--------------------|
| <b>CFIELDNAME</b> | Currency 단위를 참고하는 필드명      | 단위와 함께 값을 보여준다.    |
| <b>CHECKBOX</b>   | 체크박스로 보여줌                  | 컬럼 output 옵션       |
| <b>COL_POS</b>    | 컬럼의 output 순서              | 컬럼 output 옵션       |
| <b>COLDDICTXT</b> | Header의 라벨 설정(L,M,S,R)     | Texts              |
| <b>COLTEXT</b>    | 컬럼 라벨 TEXT                 | Texts              |
| <b>CURRENCY</b>   | Currency 단위                | 단위와 함께 값을 보여준다.    |
| <b>DD_OUTLEN</b>  | Output 길이(Characters)      | DDIC 를 참고하지 않음     |
| <b>DECIMALS_O</b> | 소수점 자리수 정의                 | 컬럼 값의 포맷           |
| <b>DECMLFIELD</b> | Decimal 필드 정의              | 컬럼 값의 포맷           |
| <b>DO_SUM</b>     | 합계 표시                      | 컬럼 output 옵션       |
| <b>DRAGDROPID</b> | Drag & Drop 용도             | Other Fields       |
| <b>EDIT_MASK</b>  | 데이터 포맷 변경                  | 컬럼 값의 포맷           |
| <b>EMPHASIZE</b>  | 컬럼 색상 강조                   | 컬럼 output 옵션       |
| <b>EXPONENT</b>   | 부동표현에 대한 지수                | 컬럼 값의 포맷           |
| <b>FIELDNAME</b>  | 내부테이블필드의 필드이름              | Output 테이블 필드      |
| <b>HOTSPOT</b>    | Single-click 에 반응          | Columns의 output 옵션 |
| <b>ICON</b>       | Icon으로 보여줌                 | 컬럼 값의 포맷           |
| <b>INTLEN</b>     | 내부길이 (바이트단위)               | DDIC 를 참고하지 않음     |
| <b>INTTYPE</b>    | ABAP data type (C,D,N,...) | DDIC 를 참고하지 않음     |
| <b>JUST</b>       | 정렬(L, R, C)                | 컬럼 값의 포맷           |
| <b>KEY</b>        | Key 필드                     | 컬럼 output 옵션       |
| <b>LOWERCASE</b>  | 소문자 사용/금지                  | 컬럼 output 옵션       |
| <b>LZERO</b>      | 선행에 제로 출력여부                | 컬럼 값의 포맷           |
| <b>NO_OUT</b>     | 필드 숨김                      | 컬럼 output 옵션       |
| <b>NO_SIGN</b>    | 출력부호 제거                    | 컬럼 값의 포맷           |
| <b>NO_SUM</b>     | 열값에 관한 합계처리하지 않음           | 컬럼 output 옵션       |
| <b>NO_ZERO</b>    | ZERO 삭제                    | 컬럼 값의 포맷           |
| <b>OUTPUTLEN</b>  | 문자의 열너비                    | Columns의 output 옵션 |
| <b>QFIELDNAME</b> | 참조한 단위필드이름                 | 단위와 함께 값을 보여준다.    |

▲ 표 18-6-1. 카타로그 속성

| 카타로그 속성    | 내역  | 사용목적               |
|------------|---|--------------------|
| QUANTITY   | 단위  | 단위와 함께 값을 보여준다.    |
| REF_FIELD  | 내부테이블 필드에 대한 참조필드이름   | Data Dictionary 참고 |
| REF_TABLE  | 내부테이블 필드에 대한 참조테이블 이름                                       | Data Dictionary 참고 |
| REPREP     | Property is selection criterion for report/report interface | Other Fields       |
| REPTEXT    | Data Element의 text  | Texts              |
| ROLLNAME   | F1 도움말을 위한 데이터 요소   | DDIC 를 참고하지 않음     |
| ROUND      | ROUND 값   | 컬럼 값의 포맷           |
| ROUNDFIELD | ROUND 특성을 가진 필드이름   | 컬럼 값의 포맷           |
| SCRTEXT_L  | 긴 필드라벨(40byte)  | Texts              |
| SCRTEXT_M  | 중간 필드라벨(20byte)   | Texts              |
| SCRTEXT_S  | 짧은 필드라벨(10byte)   | Texts              |
| SELDDICTXT | DDIC 텍스트 참조 결정  | Texts              |
| SELTEXT    | 다이얼로그 기능에 대한 열식별자   | Texts              |
| SP_GROUP   | 그룹키   | Other Fields       |
| SYMBOL     | 기호로 출력  | 컬럼 값의 포맷           |
| TECH       | Layout설정에서도 필드가 보이지 않게 함.                                   | 컬럼 output 옵션       |
| TIPDDICTXT | DDIC 텍스트 참조 결정  | Texts              |
| TOOLTIP    | 열 헤더에 대한 툴조언  | Texts              |
| TXT_FIELD  | 내부테이블필드의 필드이름   | Other Fields       |

▲ 표 18-6-1. 카타로그 속성

## 인터널 테이블과 필드카타로그

필드카타로그 정의하는 3가지 방법에 대해서 알아보았다. 이외에도 인터널 테이블의 구조를 그대로 필드카타로그로 만들수 있으며(실무에서 가장 많이 사용) 이때 인터널 테이블은 아래와 같이(구식방식) 선언된 경우에만 사용가능 하다. 인터널 테이블의 여러 테이블의 필드를 필요로 한 경우 프로그램내에서 사용하기에 아주 편리하다. [18-6-3]예제는 인터널 테이블의 필드를 그대로 필드카타로그로 구성하여 PRICE 필드의 카타로그 속성만 변경하는 것을 보여주고 있다.

예제 18-6-3

```
REPORT z18_019 .
~~
DATA : GT_SFLIGHT LIKE SFLIGHT OCCURS 0
WITH HEADER LINE.

*DATA : BEGIN OF GT_SFLIGHT OCCURS 0.
*      INCLUDE STRUCTURE SFLIGHT.
*DATA : END OF SFLIGHT.

.
~~

PERFORM getting_catalog.
PERFORM setting_catalog.
```

```
FORM getting_catalog .
DATA : LT_FIELD CAT TYPE KKBLO_T_FIELD CAT.
CALL FUNCTION 'K_KKB_FIELD CAT_MERGE'
EXPORTING
    i_tabname                = 'GT_SFLIGHT'
CHANGING
    ct_fieldcat              = lt_fieldcat[]
.
IF SY-SUBRC EQ 0.
CALL FUNCTION 'LVC_TRANSFER_FROM_KKBLO'
EXPORTING
    it_fieldcat_kkblo = lt_fieldcat[]
IMPORTING
    et_fieldcat_lvc   = gt_fieldcat[]
ENDIF.
ENDFORM.
```

```
FORM setting_catalog .
DATA ls_fieldcat TYPE lvc_s_fcat.
LOOP AT GT_FIELD CAT INTO LS_FIELD CAT.
IF ls_fieldcat-fieldname = 'PRICE'.
ls_fieldcat-coltext = 'AIR PRICE'.
ls_fieldcat-just = 'C'.
ls_fieldcat-EMPHASIZE = 'X'.
MODIFY GT_FIELD CAT FROM LS_FIELD CAT.
ENDIF.
ENDLOOP.
ENDFORM.
setting_catalog
```

결과 18-6-3

Internal Table and Field Catalog TEST

| Client | ID | No. | Flight Date | AIR PRICE | Curr. | Plane type | Capacity | Occupied | Booking total | Capacity | Occupied | Capacity |
|--------|----|-----|-------------|-----------|-------|------------|----------|----------|---------------|----------|----------|----------|
| 800    | AA | 17  | 2006.03.19  |           |       |            | 0        | 0        |               | 4        | 0        | 0        |
| 800    |    | 17  | 2006.07.19  | 1,111.00  |       |            | 0        | 0        |               | 4        | 0        | 0        |
| 800    |    | 17  | 2006.07.23  | 513.69    | USD   | 747-400    | 385      | 343      | 174,120.69    | 0        | 0        | 0        |
| 800    |    | 17  | 2006.11.05  | 513.69    | USD   | 747-400    | 385      | 335      | 169,862.28    | 0        | 0        | 0        |
| 800    |    | 17  | 2007.01.28  | 513.69    | USD   | 747-400    | 385      | 120      | 67,278.12     | 0        | 0        | 0        |
| 800    |    | 17  | 2007.01.29  |           |       |            | 0        | 0        |               | 4        | 0        | 0        |
| 800    |    | 17  | 2007.04.01  | 513.69    | USD   | 747-400    | 385      | 42       | 28,730.75     | 0        | 0        | 0        |
| 800    |    | 17  | 2007.05.13  | 513.69    | USD   | 747-400    | 385      | 9        | 13,176.20     | 0        | 0        | 0        |
| 800    |    | 17  | 2007.06.03  | 513.69    | USD   | 747-400    | 385      | 11-      | 3,354.39      | 0        | 0        | 0        |
| 800    |    | 64  | 2006.03.19  | 513.69    | USD   | 747-400    | 385      | 338      | 171,655.07    | 0        | 0        | 0        |
| 800    |    | 64  | 2006.07.23  | 513.69    | USD   | 747-400    | 385      | 343      | 172,939.30    | 0        | 0        | 0        |
| 800    |    | 64  | 2006.11.05  | 513.69    | USD   | 747-400    | 385      | 345      | 175,497.46    | 0        | 0        | 0        |
| 800    |    | 64  | 2007.01.28  | 513.69    | USD   | 747-400    | 385      | 133      | 72,877.29     | 0        | 0        | 0        |
| 800    |    | 64  | 2007.04.01  | 513.69    | USD   | 747-400    | 385      | 73       | 44,259.63     | 0        | 0        | 0        |
| 800    |    | 64  | 2007.05.13  | 513.69    | USD   | 747-400    | 385      | 0        | 8,624.89      | 0        | 0        | 0        |
| 800    |    | 64  | 2007.06.03  | 513.69    | USD   | 747-400    | 385      | 14-      | 1,643.80      | 0        | 0        | 0        |
| 800    | AZ | 555 | 2006.03.18  | 360,202   | ITL   | A319       | 220      | 189      | 69,670,298    | 0        | 0        | 0        |
| 800    |    | 555 | 2006.07.22  | 360,202   | ITL   | A319       | 220      | 178      | 66,327,620    | 0        | 0        | 0        |
| 800    |    | 555 | 2006.11.04  | 360,202   | ITL   | A319       | 220      | 191      | 70,498,762    | 0        | 0        | 0        |
| 800    |    | 555 | 2007.01.27  | 360,202   | ITL   | A319       | 220      | 88       | 36,103,058    | 0        | 0        | 0        |
| 800    |    | 555 | 2007.03.31  | 360,202   | ITL   | A319       | 220      | 45       | 21,637,340    | 0        | 0        | 0        |
| 800    |    | 555 | 2007.05.12  | 360,202   | ITL   | A319       | 220      | 10       | 9,581,376     | 0        | 0        | 0        |

## 7. ALV GRID 요소

ALV GRID 요소(Element)에 대해서 알아보자.

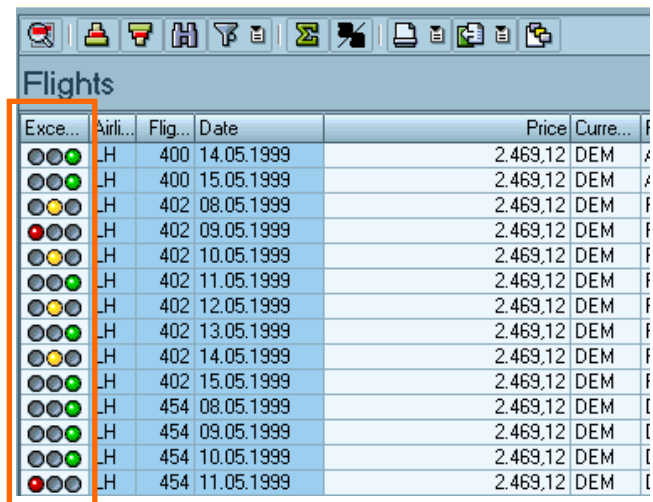
- [Output of Exceptions](#) : Grid를 신호등 표시로 보여준다.
- [Coloring Rows](#) : Grid의 row 색상을 지정한다
- [Coloring Cells](#) : Grid의 cell 색상을 지정한다.
- [Displaying Cells as Pushbuttons](#) : Cell을 Push 버튼으로 보이게 한다.

### 7-1. 신호등(Excetptions) 처리

Exceptions은 경계값을 가지는 필드를 구간에 따라서 그래픽하게 의미를 부여하게 한다. [그림 18-7-1]에서 신호등 아이콘으로 보여지며 특정필드의 값에 따라 색상을 변경되게 된다.

이것은 최종 사용자에게 데이터의 긴급성등에 대하여 쉽게 인식하게 해준다. 예를들어 재고 관리에서 있어서 안전재고의 수전은 green, 안전재고의 위험수준은 yellow, 안전재고를 초과하게 되면 red 신호등으로 보여줄수 있다.

[표 18-7-1]은 항공예약에서의 예와 함께 exceptions 필드가 가지는 내부값을 보여주고 있다.



| Exce... | Airli... | Flig... | Date       | Price    | Curre... |
|---------|----------|---------|------------|----------|----------|
| ●●●     | LH       | 400     | 14.05.1999 | 2.469,12 | DEM      |
| ●●●     | LH       | 400     | 15.05.1999 | 2.469,12 | DEM      |
| ●●●     | LH       | 402     | 08.05.1999 | 2.469,12 | DEM      |
| ●●●     | LH       | 402     | 09.05.1999 | 2.469,12 | DEM      |
| ●●●     | LH       | 402     | 10.05.1999 | 2.469,12 | DEM      |
| ●●●     | LH       | 402     | 11.05.1999 | 2.469,12 | DEM      |
| ●●●     | LH       | 402     | 12.05.1999 | 2.469,12 | DEM      |
| ●●●     | LH       | 402     | 13.05.1999 | 2.469,12 | DEM      |
| ●●●     | LH       | 402     | 14.05.1999 | 2.469,12 | DEM      |
| ●●●     | LH       | 402     | 15.05.1999 | 2.469,12 | DEM      |
| ●●●     | LH       | 454     | 08.05.1999 | 2.469,12 | DEM      |
| ●●●     | LH       | 454     | 09.05.1999 | 2.469,12 | DEM      |
| ●●●     | LH       | 454     | 10.05.1999 | 2.469,12 | DEM      |
| ●●●     | LH       | 454     | 11.05.1999 | 2.469,12 | DEM      |

▲ 그림 18-7-1. ALV Grid 신호등 처리

| Display | 내부값 | 사용 예(항공예약)         |
|---------|-----|--------------------|
| ●●●     | 3   | 좌석이 많음             |
| ●●●     | 2   | 좌석이 90% 이상 예약된 상태임 |
| ●●●     | 1   | 예약 가능 좌석이 없음       |

▲ 그림 18-7-1. Excetipons의 내부값

신호등 필드를 추가하기 위해 인터널 테이블을 다음과 같은 방법으로 선언하자.

```
DATA: BEGIN OF GT_OUTTAB OCCURS 0.
      INCLUDE STRUCTURE <DDIC-Struktur>.
DATA: light TYPE C. "신호등 표시로 보이기 위한 필드
DATA: END OF GT_OUTTAB.
```

예제 18-7-1

```
REPORT z18_020.
~~
DATA: BEGIN OF gt_sflight OCCURS 0.
      INCLUDE STRUCTURE sflight.
1 DATA: light TYPE c. "to display exceptions
DATA: END OF gt_sflight.
3
LOOP AT gt_sflight.
  if gt_sflight-SEATSOCC <= 0.
    gt_sflight-light = '1'.
  elseif gt_sflight-SEATSOCC <= 50.
    gt_sflight-light = '2'.
  else.
    gt_sflight-light = '3'.
  endif.
  MODIFY gt_sflight.
ENDLOOP.
~~
FORM setting_layout .
2
  gs_layout-excp_fname = 'LIGHT'.
ENDFORM.
```

예제 18-7-1

| Exception | Client | ID | No. | Flight Date | Curr. | Plane type | Capacity | Occupied | E |
|-----------|--------|----|-----|-------------|-------|------------|----------|----------|---|
| 000       | 800    | AA | 17  | 2006.03.19  |       |            | 0        | 0        |   |
| 000       | 800    |    | 17  | 2006.07.19  |       |            | 0        | 0        |   |
| 000       | 800    |    | 17  | 2006.07.23  | USD   | 747-400    | 385      | 343      |   |
| 000       | 800    |    | 17  | 2006.11.05  | USD   | 747-400    | 385      | 335      |   |
| 000       | 800    |    | 17  | 2007.01.28  | USD   | 747-400    | 385      | 120      |   |
| 000       | 800    |    | 17  | 2007.01.29  |       |            | 0        | 0        |   |
| 000       | 800    |    | 17  | 2007.04.01  | USD   | 747-400    | 385      | 42       |   |
| 000       | 800    |    | 17  | 2007.05.13  | USD   | 747-400    | 385      | 9        |   |
| 000       | 800    |    | 17  | 2007.06.03  | USD   | 747-400    | 385      | 11-      |   |
| 000       | 800    |    | 64  | 2006.03.19  | USD   | 747-400    | 385      | 338      |   |
| 000       | 800    |    | 64  | 2006.07.23  | USD   | 747-400    | 385      | 343      |   |
| 000       | 800    |    | 64  | 2006.11.05  | USD   | 747-400    | 385      | 345      |   |
| 000       | 800    |    | 64  | 2007.01.28  | USD   | 747-400    | 385      | 133      |   |
| 000       | 800    |    | 64  | 2007.04.01  | USD   | 747-400    | 385      | 73       |   |
| 000       | 800    |    | 64  | 2007.05.13  | USD   | 747-400    | 385      | 0        |   |
| 000       | 800    |    | 64  | 2007.06.03  | USD   | 747-400    | 385      | 14       |   |
| 000       | 800    | AZ | 555 | 2006.03.18  | ITL   | A319       | 220      | 189      |   |
| 000       | 800    |    | 555 | 2006.07.22  | ITL   | A319       | 220      | 178      |   |
| 000       | 800    |    | 555 | 2006.11.04  | ITL   | A319       | 220      | 191      |   |
| 000       | 800    |    | 555 | 2007.01.27  | ITL   | A319       | 220      | 88       |   |
| 000       | 800    |    | 555 | 2007.03.31  | ITL   | A319       | 220      | 45       |   |
| 000       | 800    |    | 555 | 2007.05.12  | ITL   | A319       | 220      | 10       |   |
| 000       | 800    |    | 555 | 2007.06.03  | ITL   | A319       | 220      | 10       |   |

### 1.Exceptions 필드 추가

인터널 테이블(OUTPUT)에 신호등 표시를 할 필드를 TYPE C로 선언하여 추가한다.

### 2.Exceptions 필드 설정

ALV 레이아웃 설정에서 신호등 필드를 추가한다.

### 3.예약석에 따라 신호등 색깔

0석보다 작으면 빨간색, 50석보다 작으면 Yellow, 그외에는 Green색으로 보여지도록 인터널 테이블 데이터를 변경한다.

## 7-2. Coloring Rows

ALV Grid 에서 강조하고 싶은 row의 색상을 변경할 수 있다. 인터널 테이블에 linecolor 필드를 추가하자.

```
DATA: BEGIN OF GT_OUTTAB OCCURS 0.
    INCLUDE STRUCTURE <DDIC-Struktur>.
DATA: linecolor(4) type c. "Color for corresponding line
DATA: END OF GT_OUTTAB.
```

예제 18-7-2

```
REPORT z18_021.
```

```
~~
```

```
DATA: BEGIN OF gt_sflight OCCURS 0.
    INCLUDE STRUCTURE sflight.
```

① DATA: linecolor(4) type c.

```
DATA: END OF gt_sflight.
```

```
~~
```

```
LOOP AT gt_sflight.
```

```
CASE GT_SFLIGHT-CARRID .
```

```
WHEN 'AA'.
```

③ GT\_SFLIGHT-linecolor = 'C100'. " Blue.

```
WHEN 'AZ'.
```

```
GT_SFLIGHT-linecolor = 'C300'. " Yellow.
```

```
WHEN 'DL'.
```

```
GT_SFLIGHT-linecolor = 'C500'. " Green.
```

```
ENDCASE.
```

```
MODIFY gt_sflight.
```

```
ENDLOOP.
```

```
~~
```

```
FORM setting_layout .
```

```
~~
```

```
* coloring row
```

② gs\_layout-INFO\_FNAME = 'LINECOLOR'.

```
ENDFORM.
```

예제 18-7-2

Coloring Row TEST

| Exception | Client | ID | No. | Flight Date | Curr. | Plane type | Capacity | Occupied |
|-----------|--------|----|-----|-------------|-------|------------|----------|----------|
|           | 800    | AA | 17  | 2006.03.19  |       |            | 0        | 0        |
|           | 800    | AA | 17  | 2006.07.19  |       |            | 0        | 0        |
|           | 800    | AA | 17  | 2006.07.23  | USD   | 747-400    | 385      | 343      |
|           | 800    | AA | 17  | 2006.11.05  | USD   | 747-400    | 385      | 335      |
|           | 800    | AA | 17  | 2007.01.28  | USD   | 747-400    | 385      | 120      |
|           | 800    | AA | 17  | 2007.01.29  |       |            | 0        | 0        |
|           | 800    | AA | 17  | 2007.04.01  | USD   | 747-400    | 385      | 42       |
|           | 800    | AA | 17  | 2007.05.13  | USD   | 747-400    | 385      | 9        |
|           | 800    | AA | 17  | 2007.06.03  | USD   | 747-400    | 385      | 11-      |
|           | 800    | AA | 64  | 2006.03.19  | USD   | 747-400    | 385      | 338      |
|           | 800    | AA | 64  | 2006.07.23  | USD   | 747-400    | 385      | 343      |
|           | 800    | AA | 64  | 2006.11.05  | USD   | 747-400    | 385      | 345      |
|           | 800    | AA | 64  | 2007.01.28  | USD   | 747-400    | 385      | 133      |
|           | 800    | AA | 64  | 2007.04.01  | USD   | 747-400    | 385      | 73       |
|           | 800    | AA | 64  | 2007.05.13  | USD   | 747-400    | 385      | 0        |
|           | 800    | AA | 64  | 2007.06.03  | USD   | 747-400    | 385      | 14       |
|           | 800    | AZ | 555 | 2006.03.18  | ITL   | A319       | 220      | 189      |
|           | 800    | AZ | 555 | 2006.07.22  | ITL   | A319       | 220      | 178      |
|           | 800    | AZ | 555 | 2006.11.04  | ITL   | A319       | 220      | 191      |
|           | 800    | AZ | 555 | 2007.01.27  | ITL   | A319       | 220      | 88       |
|           | 800    | AZ | 555 | 2007.03.31  | ITL   | A319       | 220      | 45       |
|           | 800    | AZ | 555 | 2007.05.12  | ITL   | A319       | 220      | 10       |
|           | 800    | AZ | 555 | 2007.06.03  | ITL   | A319       | 220      | 10       |

### 1. LINE COLOR 필드 추가

인터널 테이블(OUTPUT)에 line color 표시를 할 필드를 TYPE C로 선언하여 추가한다.

### 2. INFOR\_FNAME 필드 설정

ALV 레이아웃 설정에서 컬러지정 필드를 설정한다.

### 3. Carrid ID에 따라 row 색깔변경

Carrier ID 가 AA 이면 BLUE, 'AZ'이면 YELLOW, 'DL'이면 GREEN 색상을 보여주도록 설정한다.

### 7-3. Coloring Cells

ALV Grid 에서 강조하고 싶은 CELL의 색상을 변경할 수 있다. 그러나 앞에서 살펴본 ROW 단위의 색상강조보다 시간이 더 많이 소요되므로 속도가 중요한 프로그램에서는 사용하지 않기를 권한다.  
LVC\_T\_SCOL 타입의 컬러 테이블을 추가하자.

```
DATA: BEGIN OF GT_OUTTAB OCCURS 0.  
      INCLUDE STRUCTURE <DDIC-Struktur>.  
DATA: CT TYPE LVC_T_SCOL. "Table for colors  
DATA: END OF GT_OUTTAB.
```

#### 예제 18-7-3

```
REPORT z18_022.  
~~  
DATA: BEGIN OF gt_sflight OCCURS 0.  
      INCLUDE STRUCTURE sflight.  
① DATA: cellcolor TYPE lvc_t_scol.  
DATA: END OF gt_sflight.  
~~  
  
FORM setting_layout .  
  
② gs_layout-ctab_fname = 'CELLCOLOR'.  
ENDFORM.  
  
~~  
  
③ FORM setting_cell .  
  
DATA : lt_color TYPE lvc_t_scol,  
      ls_color TYPE lvc_s_scol,  
      ls_fieldcat TYPE lvc_s_fcat,  
      l_mode TYPE raw4,  
      l_type(4) TYPE c,  
      index TYPE i.
```

```
LOOP AT gt_sflight.  
  index = index + 1.  
  CLEAR: lt_color[].  
  LOOP AT gt_fieldcat INTO ls_fieldcat.  
    CLEAR ls_color.  
    ls_color-fname = ls_fieldcat-fieldname.  
    ③-1 IF ls_color-fname EQ 'PLANETYPE'.  
      CASE gt_sflight-planetype.  
        WHEN '747-400'.  
          ls_color-color-col = 5.  
          ls_color-color-int = 0.  
        WHEN 'A319'.  
          ls_color-color-col = 3.  
          ls_color-color-int = 0.  
        WHEN 'A310-300'.  
          ls_color-color-col = 6.  
          ls_color-color-int = 0.  
      ENDCASE.  
    ③-2 INSERT ls_color INTO TABLE lt_color.  
  ENDIF.  
ENDLOOP.  
CLEAR: gt_sflight-cellcolor[].  
③-3 INSERT LINES OF lt_color  
      INTO TABLE gt_sflight-cellcolor.  
MODIFY gt_sflight INDEX index.  
ENDLOOP.  
ENDFORM. " SETTING_CELL
```

### 1. CELL COLOR 필드 추가

인터널 테이블(OUTPUT)에 line color 표시를 할 필드를 TYPE LVC\_T\_SCOL 로 선언하여 추가한다.

### 2. CTAB\_FNAME 필드 설정

ALV 레이아웃 설정에서 CELL에 칼러를 지정하기 위한 필드를 설정한다. CELLCOLOR는 색상이 변경되는 필드가 아니라, 아웃풋 테이블에서 색상이 변경될 셀 정보를 담고 있는 필드이다.

### 3. CELL 색상 지정

PLANTEYP에 따라서 색상을 지정할수 있는 스크립트를 추가한다.

3-1. 필드가 PLANETYPE일 경우만 색상을 변경한다. 만약 이 IF 구문을 주석처리하게 되면, ROW 전체가 색상이 변경되어 Coloring Rows와 동일한 효과를 가져오게 된다.

3-2. It\_color 인터널 테이블에 필드명과 색상정보를 추가한다.

3-3. OUTPUT 테이블의 CELLCOLOR필드에 CELL 정보와 색상 정보를 추가한다.

프로그램을 실행하면 다음과 같은 화면을 만날수 있다.

결과 18-7-3

| Exception | Client | ID | No.  | Flight Date | Curr. | Plane type | Capacity | Occupied | Booking total | Capacity | Occupied | Capacity |
|-----------|--------|----|------|-------------|-------|------------|----------|----------|---------------|----------|----------|----------|
|           | 800    | AZ | 789  | 2006.11.03  | ITL   | 747-400    | 385      | 351      | 916,747,516   | 0        | 0        | 0        |
|           | 800    |    | 789  | 2007.01.26  | ITL   | 747-400    | 385      | 5        | 58,283,674    | 0        | 0        | 0        |
|           | 800    |    | 789  | 2007.03.30  | ITL   | 747-400    | 385      | 28       | 115,553,720   | 0        | 0        | 0        |
|           | 800    |    | 789  | 2007.05.11  | ITL   | 747-400    | 385      | 18       | 0             | 0        | 0        | 0        |
|           | 800    |    | 789  | 2007.06.01  | ITL   | 747-400    | 385      | 3        | 37,824,371    | 0        | 0        | 0        |
|           | 800    |    | 790  | 2006.03.17  | ITL   | A310-300   | 280      | 239      | 610,358,071   | 0        | 0        | 0        |
|           | 800    |    | 790  | 2006.07.21  | ITL   | A310-300   | 280      | 234      | 598,833,778   | 0        | 0        | 0        |
|           | 800    |    | 790  | 2006.11.03  | ITL   | A310-300   | 280      | 247      | 626,835,783   | 0        | 0        | 0        |
|           | 800    |    | 790  | 2007.01.26  | ITL   | A310-300   | 280      | 103      | 290,483,031   | 0        | 0        | 0        |
|           | 800    |    | 790  | 2007.03.30  | ITL   | A310-300   | 280      | 5        | 54,538,227    | 0        | 0        | 0        |
|           | 800    |    | 790  | 2007.05.11  | ITL   | A310-300   | 280      | 18       | 0             | 0        | 0        | 0        |
|           | 800    |    | 790  | 2007.06.01  | ITL   | A310-300   | 280      | 18       | 0             | 0        | 0        | 0        |
|           | 800    | DL | 1699 | 2006.03.20  | USD   | A310-300   | 280      | 241      | 124,898.88    | 0        | 0        | 0        |
|           | 800    |    | 1699 | 2006.07.24  | USD   | A310-300   | 280      | 243      | 126,522.16    | 0        | 0        | 0        |
|           | 800    |    | 1699 | 2006.11.06  | USD   | A310-300   | 280      | 241      | 123,984.50    | 0        | 0        | 0        |
|           | 800    |    | 1699 | 2007.01.29  | USD   | A310-300   | 280      | 9        | 13,155.64     | 0        | 0        | 0        |
|           | 800    |    | 1699 | 2007.01.31  | USD   | A310-300   | 280      | 9        | 13,155.64     | 0        | 0        | 0        |
|           | 800    |    | 1699 | 2007.04.02  | USD   | A310-300   | 280      | 18       | 17,306.25     | 0        | 0        | 0        |
|           | 800    |    | 1699 | 2007.05.14  | USD   | A310-300   | 280      | 18       | 0.00          | 0        | 0        | 0        |
|           | 800    |    | 1699 | 2007.06.04  | USD   | A310-300   | 280      | 18       | 0.00          | 0        | 0        | 0        |
|           | 800    |    | 1984 | 2006.03.16  | USD   | A319       | 220      | 189      | 100,097.91    | 0        | 0        | 0        |
|           | 800    |    | 1984 | 2006.07.20  | USD   | A319       | 220      | 185      | 98,505.40     | 0        | 0        | 0        |
|           | 800    |    | 1984 | 2006.11.03  | USD   | A319       | 220      | 189      | 100,097.91    | 0        | 0        | 0        |



## 7-4. Cells을 Pushbuttons 으로 보이기

ALV Grid 셀을 Pushbutton 으로 나타내어 사용자가 해당 셀을 클릭하여 더 많은 정보를 이용할 수 있을 쉽게 나타낸다. 푸쉬버튼을 클릭하게 되면, ALV GRID는 **button\_click** 이벤트를 호출하게 된다.

```
DATA: BEGIN OF GT_OUTTAB OCCURS 0.  
      INCLUDE STRUCTURE <DDIC-Struktur>.  
DATA: CT TYPE LVC_T_STYL. "Table buttons  
DATA: END OF GT_OUTTAB.
```

예제 18-7-4

```
REPORT z18_023 MESSAGE-ID ZTEST.  
~~  
DATA: BEGIN OF gt_sflight OCCURS 0.  
      INCLUDE STRUCTURE sflight.  
DATA: chk TYPE c.  
① DATA: cellbtn TYPE lvc_t_styl.  
DATA: END OF gt_sflight.  
~~  
FORM setting_layout .  
② gs_layout-stylefname = 'CELLBTN'.  
ENDFORM.  
~~  
③ FORM setting_cell .  
  
DATA : lt_cellbtn TYPE lvc_t_styl,  
      ls_cellbtn TYPE lvc_s_styl,  
      ls_fieldcat TYPE lvc_s_fcat,  
      l_mode TYPE raw4,  
      l_type(4) TYPE c,  
      index TYPE i.
```

```
LOOP AT gt_sflight.  
      index = index + 1.  
      CLEAR ls_cellbtn.  
      LOOP AT gt_fieldcat INTO ls_fieldcat.  
ls_cellbtn-fieldname = ls_fieldcat-fieldname.  
      IF ls_cellbtn-fieldname EQ 'CHK'.  
          cl_gui_alv_grid=>mc_style_button.  
          cl_gui_alv_grid=>mc_style_disabled.  
      ENDIF.  
      INSERT ls_cellbtn INTO TABLE lt_cellbtn.  
      ENLOOP.  
  
      CLEAR: gt_sflight-cellcolor[ ].  
  
      INSERT LINES OF lt_cellbtn  
          INTO TABLE gt_sflight-cellbtn.  
  
      MODIFY gt_sflight INDEX index.  
      CLEAR gt_sflight.  
  
      ENLOOP.
```

### 1. Pushbuton 필드와 CEEL 스타일 필드 추가

CHK 필드는 푸쉬버튼으로 보이게 될 필드이며, CELLBTN은 PUSHBUTTON과 같은 스타일 정보를 저장하게 되는 컬럼이다.

### 2.STYLEFNAME 필드 설정

ALV 레이아웃 설정에서 STYLEFNAME을 설정 필드를 지정한다.

### 3.PUSH BUTTUN 처리

CHK 필드를 PUSHBUTTON으로 조회될 수 있도록 설정한다. Coloring Cell 로직과 유사하다.

5

```
CLASS lcl_event_receiver DEFINITION.
```

```
PUBLIC SECTION.
```

```
~~
```

```
METHODS : handle_button_click
```

```
for event button_click of cl_gui_alv_grid
```

```
importing es_col_id es_row_no.
```

```
PRIVATE SECTION.
```

```
ENDCLASS.                                "lcl_event_receiver DEFINITION
```

```
~~
```

6

```
CLASS lcl_event_receiver IMPLEMENTATION.
```

```
METHOD handle_button_click.
```

```
CLEAR gt_sflight.
```

```
READ TABLE gt_sflight INDEX es_row_no-row_id into gs_sflight.
```

```
IF sy-subrc eq 0.
```

```
MESSAGE i002 with gs_sflight-carrid 'is selected'.
```

```
ENDIF.
```

```
ENDMETHOD.                                "handle_on_entered
```

```
ENDCLASS.                                "lcl_event_receiver IMPLEMENTATION
```

```
~~
```

```
FORM setting_event .
```

4

```
CREATE OBJECT event_receiver.
```

```
SET HANDLER event_receiver->handle_button_click FOR grid1.
```

```
ENDFORM.                                " setting_event
```

#### 결과 18-7-4

Display cell as pushbutton TEST

| Exception | Client | ID | No.        | Flight Date | Capacity | Occupied   | Booking total | Capacity | Occupied | Capacity | Occupied |  |
|-----------|--------|----|------------|-------------|----------|------------|---------------|----------|----------|----------|----------|--|
| 800       | AA     | 17 | 2006.03.19 | 0           | 0        |            |               | 4        | 0        | 0        | 0        |  |
| 800       |        | 17 | 2006.07.19 | 0           | 0        |            |               | 4        | 0        | 0        | 0        |  |
| 800       |        | 17 | 2006.07.23 | 385         | 343      | 174,120.69 |               | 0        | 0        | 0        | 0        |  |
| 800       |        | 17 | 2006.11.05 | 385         | 335      | 169,862.28 |               | 0        | 0        | 0        | 0        |  |
| 800       |        | 17 | 2007.01.28 | 385         | 120      | 67,278.12  |               | 0        | 0        | 0        | 0        |  |
| 800       |        | 17 | 2007.01.29 | 0           | 0        |            |               | 4        | 0        | 0        | 0        |  |

SAP

i:ZTEST:002 AZ is selected!

| Exception | Client | ID  | No.        | Flight Date | Capacity | Occupied   | Booking total | Capacity | Occupied | Capacity | Occupied |  |
|-----------|--------|-----|------------|-------------|----------|------------|---------------|----------|----------|----------|----------|--|
| 800       | AZ     | 555 | 2006.03.18 | 220         | 189      | 69,670,298 |               | 0        | 0        | 0        | 0        |  |
| 800       |        | 555 | 2006.07.22 | 220         | 178      | 66,327,620 |               | 0        | 0        | 0        | 0        |  |
| 800       |        | 555 | 2006.11.04 | 220         | 191      | 70,498,762 |               | 0        | 0        | 0        | 0        |  |
| 800       |        | 555 | 2007.01.27 | 220         | 88       | 36,103,058 |               | 0        | 0        | 0        | 0        |  |
| 800       |        | 555 | 2007.03.31 | 220         | 45       | 21,637,340 |               | 0        | 0        | 0        | 0        |  |
| 800       |        | 555 | 2007.05.12 | 220         | 10       | 9,581,376  |               | 0        | 0        | 0        | 0        |  |
| 800       |        | 555 | 2007.05.23 | 220         | 10       |            |               | 0        | 0        | 0        | 0        |  |

#### 4. 버튼 클릭의 이벤트 등록

푸쉬 버튼을 클릭하게 되면 반응하게 되는 이벤트 핸들러 메소드를 등록한다.

#### 5. 이벤트 메소드 정의

이벤트 메소드를 정의한다.

#### 6. 이벤트 메소드 구현

버튼을 클릭하게 되면 해당 row의 carrid를 읽어 정보 메세지창을 오픈하는 스크립트를 추가한다.

## 8. Context Menu정의

Context 메뉴는 SAP GUI에서 정의된 화면의 사용자 인터페이스로서 한마디로 정의하면 사용자가 마우스 오른쪽 버튼(또는 Shift+F10)을 클릭할때 반응하는 메뉴를 Context Menu라 정의한다. 각 메뉴에는 그에 반응하는 기능이 할당되어 있다. 표준 Context 메뉴 이외에 다음의 화면 요소에 사용자 정의 메뉴를 추가할 수 있다.

- Input/output fields
- Text fields
- Table controls
- Frames
- Subscreens

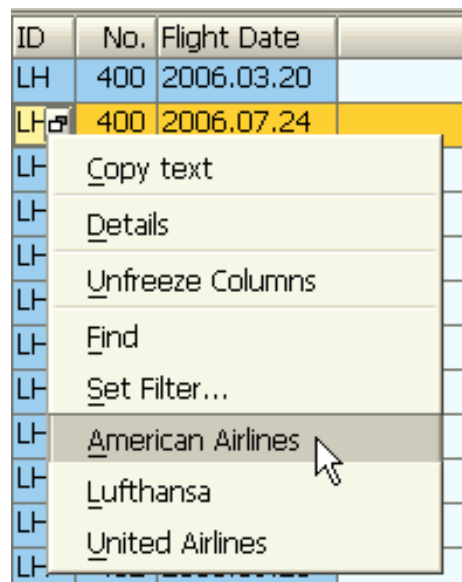
이러한 화면요소에서 마우스 오른쪽 버튼을 클릭할때, ABAP 프로그램에서 Context 메뉴를 동적으로 생성할수 있다. 메뉴 페인터를 이용해 스크린을 생성한후 Status를 할당하면 시스템이 자동으로 표준 Conext 메뉴를 제공한다. 이 표준 메뉴는 스크린의 기능키(Function Key)를 포함하게 된다. 이외의 기능은 스크립트를 추가하여야하며, 이것은 function key에 제한이 없이 어떠한 기능도 포함할 수 있다. 그러나 pushbuttons, checkboxes 그리고 radio buttons에는 Context 메뉴를 사용할수 없다.

- 이러한 메뉴들은 자신만의 기능을 포함할수 있다.-

이번 장에서는 ALV Grid 내에서 Contex 메뉴의 사용법을 설명하게 된다.

### 8-1. Context 메뉴 Class(CL\_CTMENU)

Context 메뉴는 Globa ABAP Objects 클래스인 CL\_CTMENU의 객체이다. 이 클래스 라이브러리는 Control Framework(Custom Contol)클래스에 속한다. 이것은 프로그램내에서 Context 메뉴를 동적으로 구성할수 있는 메소드를 포함하고 있다는 의미이다. CL\_CTMENU 클래스에서 가장중요한 메소드는 [표 18-7-1]과 같다.



▲ 그림 18-8-1. ALV에서의 Context 메뉴

| Method               | Function   |
|----------------------|--|
| LOAD_GUI_STATUS      | 이미 정의되어 있는 <b>Context</b> 메뉴를 프로그램내에서 선언한 <b>Context</b> 메뉴에 할당한다.(SE41 : 메뉴페인터) |
| ADD_FUNCTION         | 프로그램내에서 <b>Context</b> 메뉴에 하나의 기능을 할당한다  |
| ADD_MENU             | 프로그램내의 <b>Conext</b> 메뉴에 또다른 <b>Local Conext</b> 메뉴를 추가한다.                       |
| ADD_SUBMENU          | <b>Conext</b> 메뉴에 종속적인 <b>Local Conext</b> 메뉴를 추가한다.                             |
| ADD_SEPARATOR        | 구분자를 삽입한다.   |
| HIDE_FUNCTIONS       | 기능을 숨긴다.   |
| SHOW_FUNCTIONS       | 기능을 보인다.   |
| DISABLE_FUNCTIONS    | 기능을 비활성화 한다.   |
| ENABLE_FUNCTIONS     | 기능을 활성화한다.   |
| SET_DEFAULT_FUNCTION | <b>Conext</b> 메뉴에서 기본적으로 선택되어 보여지는 기능이다.   |

▲ 표 18-8-1. CL\_CTMENU 클래스의 주요 메소드

[표18-8-1]에서 Local Contetx 메뉴는 클래스 CL\_CTMENU의 객체를 의미한다. Context 메뉴 클래스는 구문에 따라서 다른 방법으로 사용하게 된다.

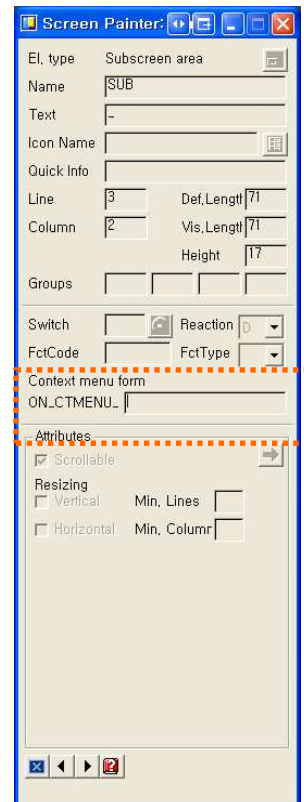
## ■ CONTROL

Contol 클래스에 이미 CL\_CTMENU 클래스가 캡슐화되어 있으므로 객체를 생성할 필요없이 사용하기만 하면된다. 일반적으로 Contol을 이용하면 Conext 메뉴를 생성하지 않는다. 이것은 Contol 자신이 포함하고 있는 Context 메뉴의 이벤트와 충돌을 피하게 한다.

## ■ 스크린(LIST)

일반 스크린에서 Context 메뉴를 정의하게 되면, CL\_CTMENU 클래스의 객체가 실행시에 자동으로 생성된다. 그러므로 프로그램내에서 명시적으로 선언할 필요가 없다. 화면의 요소에 Context 메뉴를 연결시키기 위해서는 메뉴 페인터를 이용해 Context ID를 입력하여 한다.

Context 메뉴는 계층구조를 이루고 있기때문에 Group box에 속한 화면 요소들이 Context 메뉴가 포함되어 있지 않다면 Group Box의 메뉴를 상속받게 된다.



## 8-2. Context 메뉴 생성

Context 메뉴는 메뉴 페인트(Tcode SE41)에서 생성 가능하다.

The screenshots illustrate the process of creating a Context Menu in SAP SE41:

- 1. Menu Painter: Initial Screen**: The 'Program' field is set to 'Z18\_025'. Under 'Subobjects', 'Status' is selected, and 'CONTEXT\_MENU1' is entered. The 'Create' button is highlighted.
- 2. Create Status**: The 'Status' field is 'CONTEXT\_MENU1'. Under 'Status type', 'Context Menu' is selected.
- 3. Maintain Status CONTEXT\_MENU1 of Interface Z18\_025**: The 'Context menu' table is shown with the following data:

| Code | Text    |
|------|---------|
| RESH | REFRESH |
| DETL | DETAIL  |
- 4. Maintain Status CONTEXT\_MENU1 of Interface Z18\_025**: The 'Object Name' tree on the left shows the hierarchy: Z18\_019 > Z18\_025 > GUI Status > CONTEXT\_MENU1. The 'Context menu' table is visible on the right.

### 1. Context 메뉴 생성

Context 메뉴를 추가할 프로그램명, Context 메뉴명을 입력하고 생성 버튼을 클릭한다.

### 2. Context Menu 선택

Dialog Box에서 Context Menu를 선택하고, 엔터를 입력한다.

### 3. Function code 추가

Context 메뉴에 추가하고자 하는 Function Code를 입력하고 활성화 버튼을 클릭한다.

### 4. Context Menu 확인

프로그램 GUI Status에서 Context 메뉴가 생성되었는지 확인한다.

### 8-3. ALV에서 Context 메뉴 추가

스크린 페인터를 이용해 Custmer Contol을 오픈한다.  
Contol에는 Contet 메뉴를 입력하는 부분이 비활성화되어 있다.  
ALV Grid에 이벤트를 추가하여, Context 메뉴를 추가해주어야한다.

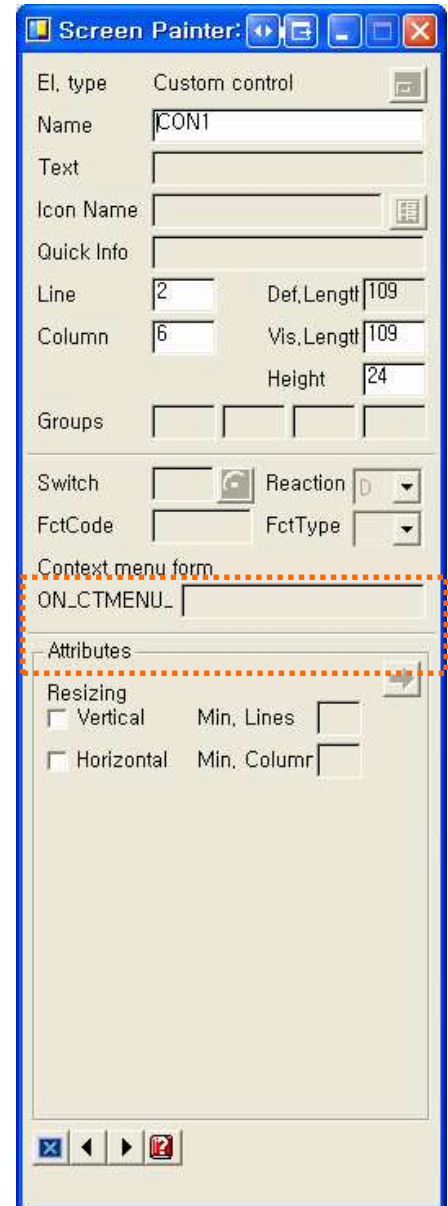
예제 18-7-1

```
REPORT z18_025 .
    ~~
CREATE OBJECT g_application.
    SET HANDLER event_receiver
        ->handle_context_menu FOR grid1.
    ~~
CLASS lcl_event_receiver DEFINITION.

    METHODS : handle_context_menu
        FOR EVENT context_menu_request OF cl_gui_alv_grid
            IMPORTING e_object.

ENDCLASS.
    ~~
CLASS lcl_event_receiver IMPLEMENTATION.
    call method e_object->LOAD_GUI_STATUS
        exporting PROGRAM = SY-REPID
                STATUS = 'CONTEXT_MENU1'
                MENU = e_object.

    ENDMETHOD.
ENDCLASS.
    ~~
```



▲ 그림 18-7-2. Contol과 Context 메뉴

[예제 18-7-1]에서는 LOAD\_GUI\_STATUS 메소드를 이용해 CONTEXT 메뉴를 추가하는 스크립트를 보여준다. 앞의 예제에서 이미 구현된 이벤트 클래스내에서 이벤트 핸들러 메소드를 등록해주고, 메소드를 구현하면 된다. 프로그램을 실행하여 보자.

## 결과 18-8-1

| Carid | Flight Numb | Flight Date | Airfare  | Curr. | Plane type | Capaci. | Occupied | Booking total | Capaci. | Occupied | Capa. |
|-------|-------------|-------------|----------|-------|------------|---------|----------|---------------|---------|----------|-------|
| AA    | 17          | 2006.03.19  |          |       |            | 0       | 0        |               | 4       | 0        |       |
| AA    | 17          | 2006.07.19  | 1,111.00 |       |            | 0       | 0        |               | 4       | 0        |       |
| AA    | 17          | 2006.07.23  | 513.69   | USD   | 747-400    | 385     | 343      | 174,120.69    | 0       | 0        |       |
| AA    | 17          | 2006.11.05  | 513.69   | USD   | 747-400    | 385     | 335      | 169,862.28    | 0       | 0        |       |
| AA    | 17          | 2007.01.28  | 513.69   | USD   | 747-400    | 385     | 120      | 67,278.12     | 0       | 0        |       |
| AA    | 17          | 2007.01.29  |          |       |            | 0       | 0        |               | 4       | 0        |       |
| AA    | 17          | 2007.04.01  | 513.69   | USD   | 747-400    | 385     | 42       | 28,730.75     | 0       | 0        |       |
| AA    | 17          | 2007.05.13  | 51       |       |            | 85      | 9        | 13,176.20     | 0       | 0        |       |
| AA    | 17          | 2007.06.03  | 51       |       |            | 85      | 11       | 3,354.39      | 0       | 0        |       |
| AA    | 64          | 2006.03.19  | 51       |       |            | 85      | 338      | 171,655.07    | 0       | 0        |       |
| AA    | 64          | 2006.07.23  | 51       |       |            | 85      | 343      | 172,939.30    | 0       | 0        |       |
| AA    | 64          | 2006.11.05  | 51       |       |            | 85      | 345      | 175,497.46    | 0       | 0        |       |
| AA    | 64          | 2007.01.28  | 51       |       |            | 85      | 139      | 72,877.29     | 0       | 0        |       |
| AA    | 64          | 2007.04.01  | 51       |       |            | 85      | 73       | 44,299.63     | 0       | 0        |       |
| AA    | 64          | 2007.05.13  | 51       |       |            | 85      | 0        | 8,624.89      | 0       | 0        |       |
| AA    | 64          | 2007.06.03  | 51       |       |            | 85      | 14       | 1,643.80      | 0       | 0        |       |
| AZ    | 555         | 2006.03.18  | 360,202  | ITL   | A319       | 220     | 189      | 69,670.298    | 0       | 0        |       |
| AZ    | 555         | 2006.07.22  | 360,202  | ITL   | A319       | 220     | 178      | 66,327.620    | 0       | 0        |       |
| AZ    | 555         | 2006.11.04  | 360,202  | ITL   | A319       | 220     | 191      | 70,496.762    | 0       | 0        |       |
| AZ    | 555         | 2007.01.27  | 360,202  | ITL   | A319       | 220     | 88       | 36,103.058    | 0       | 0        |       |
| AZ    | 555         | 2007.03.31  | 360,202  | ITL   | A319       | 220     | 45       | 21,637.340    | 0       | 0        |       |
| AZ    | 555         | 2007.05.12  | 360,202  | ITL   | A319       | 220     | 10       | 9,581.376     | 0       | 0        |       |
| AZ    | 555         | 2007.06.05  | 360,202  | ITL   | A319       | 220     | 10       | 9,581.376     | 0       | 0        |       |

[결과 18-8-1] 화면에서 볼 수 있듯이 REFRESH와 DETAIL Function 코드가 추가되었다. CL\_CTMENU 클래스의 객체를 생성하지 않고, 바로 Context 메뉴를 load 할 수 있는 것은 [그림 18-7-3]의 CL\_GUI\_ALV\_GRID 내에 Context 메뉴에 관련된 컴포넌트가 추가되어 있기 때문이다.

REFRESH 평성 코드를 클릭하게 되면 화면이 REFRESH 된다. [예제 18-4-6]에서 이미 REFRESH 명령에 대한 스크립트를 추가해두었기 때문이다.

▼ 그림 18-8-3. CL\_GUI\_ALV\_GRID와 CONTEXT 메뉴

### Class Builder: Display Class CL\_GUI\_ALV\_GRID

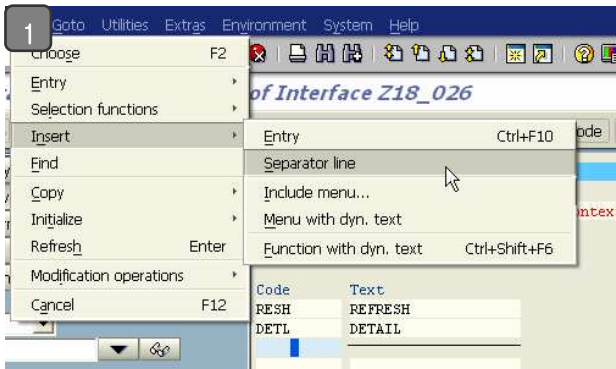
| Event                | Type    | Visi  | Mo                       | Description  |
|----------------------|---------|-------|--------------------------|--------------|
| CONTEXT MENU REQUEST | Instanc | Publi | <input type="checkbox"/> | Context Menu |
| MENU_BUTTON          | Instanc | Publi | <input type="checkbox"/> | Menu Button  |
| TOOLBAR              | Instanc | Publi | <input type="checkbox"/> | Toolbar      |

### Class Builder: Display Class CL\_GUI\_ALV\_GRID

| Methods               | Level   | Visi | Mo                       | M | Description                    |
|-----------------------|---------|------|--------------------------|---|--------------------------------|
| INIT_CONTEXT_MENU     | Instanc | Priv | <input type="checkbox"/> |   | Define Right Mouse Button Menu |
| INIT_TOOLBAR          | Instanc | Priv | <input type="checkbox"/> |   | Initialize Toolbar Status      |
| LAYOUT_ADMINISTRATION | Instanc | Priv | <input type="checkbox"/> |   | Layout Administration          |

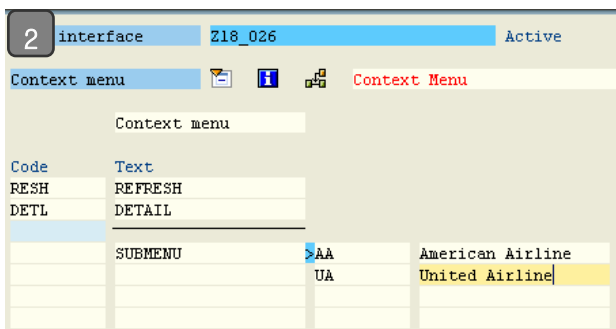
## 8-4. Context 서브 메뉴 생성

Context 메뉴에 서브 메뉴를 추가해보자.



### 1. 구분자 추가

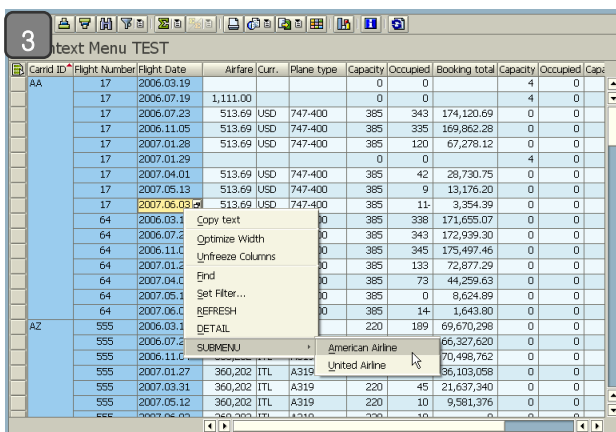
MENU의 EDIT- -> INSERT -> Separator Line 을 선택하여 구분자를 삽입하자.



### 2. SUB MENU 추가

Code 필드에는 아무값도 입력하지 않고, Text 에만 메뉴명을 입력하고 오른쪽에 생성되는 노드에 Sub 메뉴를 추가하자.

서브 메뉴 AA를 선택하면 Carrier ID 가 'AA' 인 것만 조회되도록 프로그램을 하기 위한 선행과정이다.



### 3. 프로그램 실행

[예제 18-7-1]을 Z18\_0026 프로그램으로 복사 생성하여 Context 메뉴만 변경하여 실행한 화면이다. 위에서 설정한 구분선과 서브 메뉴를 확인할 수 있다.



## 8-5. Context 메뉴의 메소드 실습

이제까지는 메뉴 페인터를 이용해 Context 메뉴를 구성하는 방법을 학습하였다. 이와 같은 방법 외에도 메뉴 클래스의 메소드를 통해 Function code를 추가하고 이미 생성된 메뉴코드를 숨기는 기능 등을 할 수 있다. [표 18-7-1]의 메소드들에 대해서 실습해보자.

예제 18-8-2

REPORT z18\_026.

METHOD handle\_context\_menu.

DATA: lt\_std\_fcodes TYPE ui\_functions.

DATA: lt\_own\_fcodes TYPE ui\_functions.

CALL METHOD e\_object->load\_gui\_status

EXPORTING

program = sy-repid

status = 'CONTEXT\_MENU1'

menu = e\_object.

CALL METHOD e\_object->add\_function

EXPORTING

fcode = 'LEAVE'

text = 'LEAVE PROGRAM'.

APPEND cl\_gui\_alv\_grid=>mc\_fc\_col\_optimize  
TO lt\_std\_fcodes.

CALL METHOD e\_object->hide\_functions

EXPORTING

fcodes = lt\_std\_fcodes.

APPEND 'RESH' TO lt\_own\_fcodes.

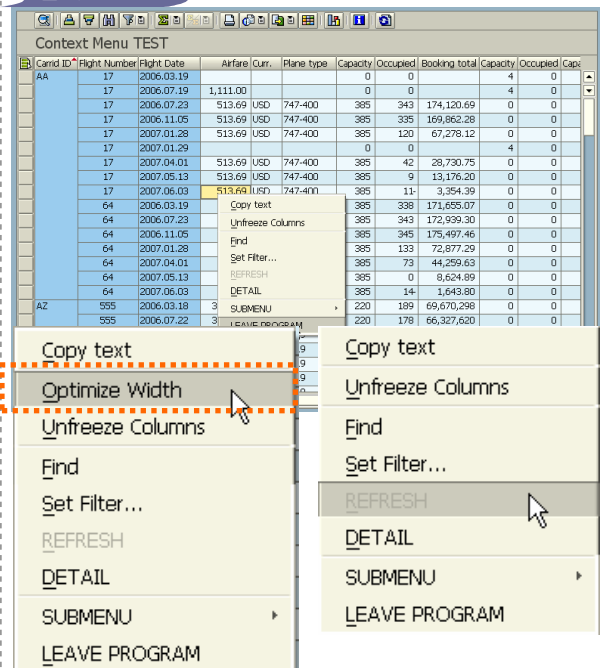
CALL METHOD e\_object->disable\_functions

EXPORTING

fcodes = lt\_own\_fcodes.

ENDMETHOD.

결과 18-8-2



### 1. add\_function 메소드

add\_function 메소드를 통해 Context 메뉴에 Function Code를 추가한다.

### 2. hide\_functions

Context 메뉴에 function code를 숨기게 한다. (Show\_functions는 반대역할)

### 3. disable\_functions.

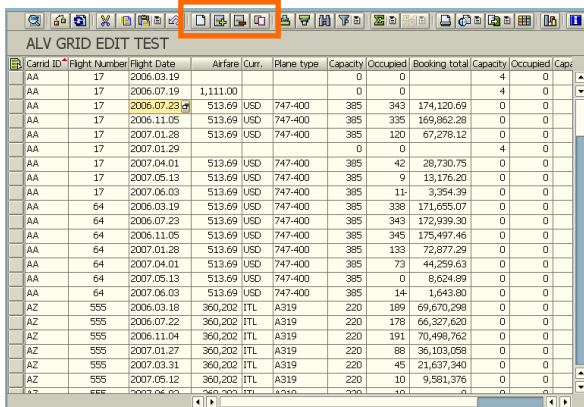
Context 메뉴의 function 코드를 disable 시킨다. (Enable\_ffunctions는 반대역할)

### 4. function code의 인터널 테이블

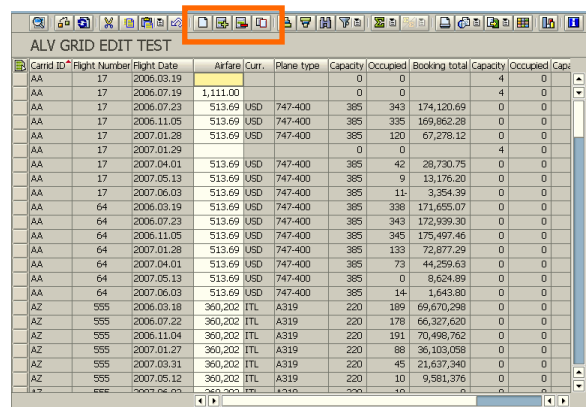
ui\_functions 타입의 인터널 테이블이며 Context 메뉴의 function code들을 추가하여 스크립트를 통해 메소드에서 활용할 수 있도록 한다.

## 9. ALV GRID EDIT

모듈 풀 프로그램에서 테이블 컨트롤 데이터를 자유롭게 변경하는 것처럼 ALV GRID 도 EDIT 기능을 지원한다. ALV의 데이터를 편집하는 방법은 크게 두가지가 존재한다. 첫째는 레이아웃 설정에서 편집 옵션을 주어 전체 GRID를 변경할수 있으며, 둘째 필드카타로그 레벨에서 편집 옵션을 이용하여 필드별로 변경 설정을 할수 있다. 키필드는 변경이 불가능하게 설정하여야 하므로 필드카타로그에서 설정하여 필드별로 변경하도록 하는 것이 바람직하다. [그림 18-9-1]의 왼쪽 그림은 전체 ALV의 필드를 수정가능하도록 하였으며, 오른쪽 그림은 PRICE 필드만 수정이 가능하다. 그리고 EDIT 속성을 활성화하면 기본적으로 TOOLBAR에 ALV ROW를 생성/변경/삭제 할수 있는 버튼이 추가된다.



| Carid ID | Flight Number | Flight Date | Airfare  | Curr. | Plane type | Capacity | Occupied | Booking total | Capacity | Occupied | Cap |
|----------|---------------|-------------|----------|-------|------------|----------|----------|---------------|----------|----------|-----|
| AA       | 17            | 2006.03.19  |          |       |            | 0        | 0        |               |          | 4        | 0   |
| AA       | 17            | 2006.07.19  | 1,111.00 |       |            |          |          |               |          |          |     |
| AA       | 17            | 2006.07.23  | 513.69   | USD   | 747-400    | 385      | 343      | 174,120.69    | 0        | 0        |     |
| AA       | 17            | 2006.11.05  | 513.69   | USD   | 747-400    | 385      | 335      | 169,862.28    | 0        | 0        |     |
| AA       | 17            | 2007.01.28  | 513.69   | USD   | 747-400    | 385      | 120      | 67,278.12     | 0        | 0        |     |
| AA       | 17            | 2007.01.29  |          |       |            | 0        | 0        |               |          | 4        | 0   |
| AA       | 17            | 2007.04.01  | 513.69   | USD   | 747-400    | 385      | 42       | 28,730.75     | 0        | 0        |     |
| AA       | 17            | 2007.05.13  | 513.69   | USD   | 747-400    | 385      | 9        | 13,176.20     | 0        | 0        |     |
| AA       | 17            | 2007.06.03  | 513.69   | USD   | 747-400    | 385      | 11       | 3,354.39      | 0        | 0        |     |
| AA       | 64            | 2006.03.19  | 513.69   | USD   | 747-400    | 385      | 338      | 171,655.07    | 0        | 0        |     |
| AA       | 64            | 2006.07.23  | 513.69   | USD   | 747-400    | 385      | 343      | 172,939.30    | 0        | 0        |     |
| AA       | 64            | 2006.11.05  | 513.69   | USD   | 747-400    | 385      | 345      | 175,497.46    | 0        | 0        |     |
| AA       | 64            | 2007.01.28  | 513.69   | USD   | 747-400    | 385      | 133      | 72,877.29     | 0        | 0        |     |
| AA       | 64            | 2007.04.01  | 513.69   | USD   | 747-400    | 385      | 73       | 44,259.63     | 0        | 0        |     |
| AA       | 64            | 2007.05.13  | 513.69   | USD   | 747-400    | 385      | 0        | 6,624.89      | 0        | 0        |     |
| AA       | 64            | 2007.06.03  | 513.69   | USD   | 747-400    | 385      | 14       | 1,643.80      | 0        | 0        |     |
| AZ       | 555           | 2006.03.18  | 360,202  | ITL   | A319       | 220      | 189      | 69,670,298    | 0        | 0        |     |
| AZ       | 555           | 2006.07.22  | 360,202  | ITL   | A319       | 220      | 178      | 66,327,620    | 0        | 0        |     |
| AZ       | 555           | 2006.11.04  | 360,202  | ITL   | A319       | 220      | 191      | 70,498,762    | 0        | 0        |     |
| AZ       | 555           | 2007.01.27  | 360,202  | ITL   | A319       | 220      | 88       | 36,103,058    | 0        | 0        |     |
| AZ       | 555           | 2007.03.31  | 360,202  | ITL   | A319       | 220      | 45       | 21,637,340    | 0        | 0        |     |
| AZ       | 555           | 2007.05.12  | 360,202  | ITL   | A319       | 220      | 10       | 9,581,376     | 0        | 0        |     |
| AZ       | 555           | 2007.06.03  | 360,202  | ITL   | A319       | 220      | 10       | 9,581,376     | 0        | 0        |     |



| Carid ID | Flight Number | Flight Date | Airfare  | Curr. | Plane type | Capacity | Occupied | Booking total | Capacity | Occupied | Cap |
|----------|---------------|-------------|----------|-------|------------|----------|----------|---------------|----------|----------|-----|
| AA       | 17            | 2006.03.19  |          |       |            | 0        | 0        |               |          | 4        | 0   |
| AA       | 17            | 2006.07.19  | 1,111.00 |       |            |          |          |               |          |          |     |
| AA       | 17            | 2006.07.23  | 513.69   | USD   | 747-400    | 385      | 343      | 174,120.69    | 0        | 0        |     |
| AA       | 17            | 2006.11.05  | 513.69   | USD   | 747-400    | 385      | 335      | 169,862.28    | 0        | 0        |     |
| AA       | 17            | 2007.01.28  | 513.69   | USD   | 747-400    | 385      | 120      | 67,278.12     | 0        | 0        |     |
| AA       | 17            | 2007.01.29  |          |       |            | 0        | 0        |               |          | 4        | 0   |
| AA       | 17            | 2007.04.01  | 513.69   | USD   | 747-400    | 385      | 42       | 28,730.75     | 0        | 0        |     |
| AA       | 17            | 2007.05.13  | 513.69   | USD   | 747-400    | 385      | 9        | 13,176.20     | 0        | 0        |     |
| AA       | 17            | 2007.06.03  | 513.69   | USD   | 747-400    | 385      | 11       | 3,354.39      | 0        | 0        |     |
| AA       | 64            | 2006.03.19  | 513.69   | USD   | 747-400    | 385      | 338      | 171,655.07    | 0        | 0        |     |
| AA       | 64            | 2006.07.23  | 513.69   | USD   | 747-400    | 385      | 343      | 172,939.30    | 0        | 0        |     |
| AA       | 64            | 2006.11.05  | 513.69   | USD   | 747-400    | 385      | 345      | 175,497.46    | 0        | 0        |     |
| AA       | 64            | 2007.01.28  | 513.69   | USD   | 747-400    | 385      | 133      | 72,877.29     | 0        | 0        |     |
| AA       | 64            | 2007.04.01  | 513.69   | USD   | 747-400    | 385      | 73       | 44,259.63     | 0        | 0        |     |
| AA       | 64            | 2007.05.13  | 513.69   | USD   | 747-400    | 385      | 0        | 6,624.89      | 0        | 0        |     |
| AA       | 64            | 2007.06.03  | 513.69   | USD   | 747-400    | 385      | 14       | 1,643.80      | 0        | 0        |     |
| AZ       | 555           | 2006.03.18  | 360,202  | ITL   | A319       | 220      | 189      | 69,670,298    | 0        | 0        |     |
| AZ       | 555           | 2006.07.22  | 360,202  | ITL   | A319       | 220      | 178      | 66,327,620    | 0        | 0        |     |
| AZ       | 555           | 2006.11.04  | 360,202  | ITL   | A319       | 220      | 191      | 70,498,762    | 0        | 0        |     |
| AZ       | 555           | 2007.01.27  | 360,202  | ITL   | A319       | 220      | 88       | 36,103,058    | 0        | 0        |     |
| AZ       | 555           | 2007.03.31  | 360,202  | ITL   | A319       | 220      | 45       | 21,637,340    | 0        | 0        |     |
| AZ       | 555           | 2007.05.12  | 360,202  | ITL   | A319       | 220      | 10       | 9,581,376     | 0        | 0        |     |
| AZ       | 555           | 2007.06.03  | 360,202  | ITL   | A319       | 220      | 10       | 9,581,376     | 0        | 0        |     |

▲ 그림 18-9-1. ALV GRID EDIT

### 1. 레이아웃 설정

먼저 Z18\_017 프로그램을 Z18\_027 로 복사 생성하여 실습해보자. 레이아웃의 EDIT 속성을 설정하면 [그림 18-9-1]그림의 왼쪽과 같이 전체 ALV가 수정가능하도록 변경된다. 레이아웃 설정만 변경한후 프로그램을 실행하며 확인해보자. 그리고 생성/변경/삭제 버튼도 클릭하여 해당 이벤트의 동작도 실습해보자.

### 2. 필드 카타로그 속성

PRICE 필드의 카타로그에 EDIT 속성을 설정하면 [그림 18-9-1]의 오른쪽 화면과 같이 해당 필드만 변경이 가능하도록 설정된다.

이때는 레이아웃의 EDIT 속성을 초기화하여야 한다.

1 REPORT z18\_027 .

FORM setting\_layout .

gs\_layout-edit = 'X'.

ENDFORM.

2 FORM setting\_catalog .

ls\_fieldcat-fieldname = 'PRICE'.


ls\_fieldcat-coltext = 'Airfare'.

ls\_fieldcat-edit = 'X'.

APPEND ls\_fieldcat TO gt\_fieldcat.

ENDFORM.

### 3. SET\_READY\_FOR\_INPUT 메소드

GRID의 EDIT 기능을 활성화하는 메소드로써, 레이아웃/필드카타로그에서 설정한 EDIT기능을 Disable 시키게 된다. 이 메소드는 ALV GRID에서  버튼을 추가하여 변경버튼을 한번 클릭하면 ALV 를 편집모드로 변경하고, 다시 클릭하면 조회모드로 변경하고자 하는 경우에 많이 사용된다.

### 4. GUI STATUS 설정

사용자가 ALV GRID의 필드 값을 변경한후 저장버튼을 클릭하면 인터널 테이블의 값(또는DB 테이블의 값)을 변경한후 ALV를 REFRESH 하면 된다.

스크린 100번에 추가해준 GUI STATUS 'G100'에 SAVE 버튼을 활성화해주자.

### 5. 저장 이벤트

스크린 100번의 PAI 모듈에 저장버튼을 클릭하였을 경우의 스크립트를 추가한다.

**check\_changed\_data** 메소드는 GRID의 필드가 포맷에 맞게 변경되면 'X'값을 반환한다. 조건에 맞게 값이 변경되었다면, PERFORM update\_database에서 DB 테이블의 데이터를 변경하게 된다.

update\_database 서브루틴은gt\_modified\_rows 라는 인터널 테이블의 내용으로 SFLIGHT 테이블을 변경하는 로직으로 구사되어 있다. 이때 **GT\_MODIFIED\_ROWS** 인터널 테이블은 SFLIGHT 테이블의 테이블로 ALV GRID의 데이터가 변경되면 변경된 정보를 담고 있는 테이블이다. 이 부분에 대한 스크립트를 다음 6단계에서 살펴보자.

3

```
CALL METHOD GRID1->SET_READY_FOR_INPUT
EXPORTING I_READY_FOR_INPUT = 1.
* Value 1 : Ready for input
* Value 0 : Not ready for input
```

4



5

```
MODULE user_command_0100 INPUT.
CASE sy-ucomm.
WHEN 'SAVE'.
DATA: l_valid TYPE c.
CALL METHOD grid1->check_changed_data
IMPORTING
e_valid = l_valid.
IF l_valid IS NOT INITIAL.
perform update_database.
ENDIF.
ENDCASE.
ENDMODULE.
```

```
FORM update_database .
MODIFY SFLIGHT FROM TABLE
GT_MODIFIED_ROWS.
IF SY-SUBRC EQ 0.
MESSAGE s000(OK) WITH 'SAVE
OK'.
ENDIF.
ENDFORM.
```

6

```
CLASS lcl_event_receiver DEFINITION.
```

```
PUBLIC SECTION.
```

```
METHODS:
```

```
    handle_data_changed
```

```
    FOR EVENT data_changed OF cl_gui_alv_grid
```

```
    IMPORTING er_data_changed.
```

```
ENDCLASS.
```

7

```
CLASS lcl_event_receiver IMPLEMENTATION.
```

```
METHOD handle_data_changed.
```

```
    DATA: ls_sflight TYPE sflight,
```

```
           ls_outtab LIKE LINE OF gt_sflight.
```

```
    FIELD-SYMBOLS: <fs> TYPE table.
```

```
    ASSIGN er_data_changed->mp_mod_rows->* TO <fs>.
```

```
    LOOP AT <fs> INTO ls_outtab.
```

```
        MOVE-CORRESPONDING ls_outtab TO ls_sflight.
```

```
        APPEND ls_sflight TO gt_modified_rows.
```

```
    ENDOLOOP.
```

```
ENDMETHOD.                                "handle_data_changed
```

8

```
DATA: g_verifier TYPE REF TO lcl_event_receiver.
```

```
CREATE OBJECT g_verifier.
```

```
SET HANDLER g_verifier->handle_data_changed FOR grid1.
```

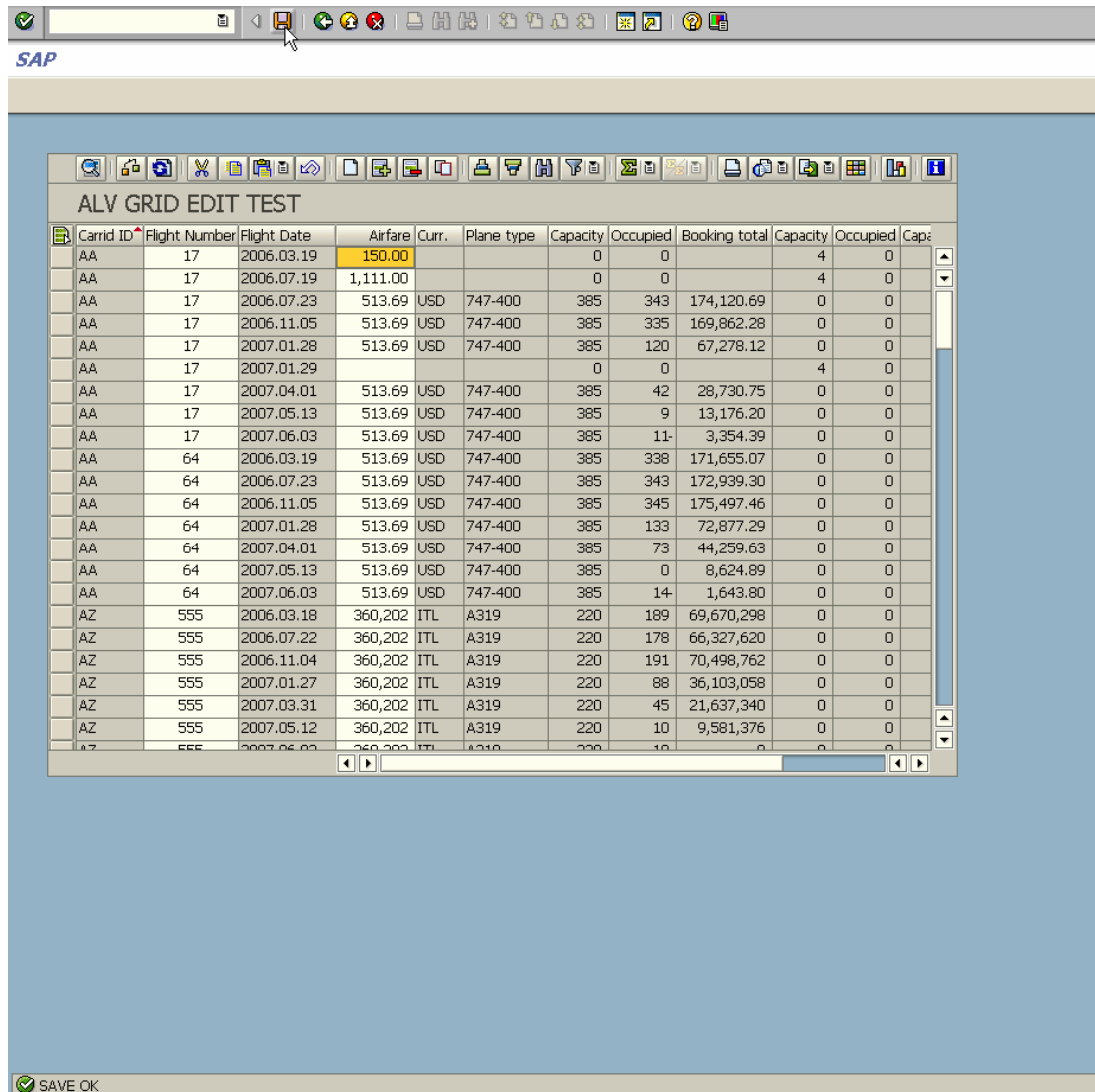
## 6. 클래스 정의 ( for 이벤트 핸들러 메소드 )

ALV GRID의 값이 변경되면 변경된 정보를 저장하기 위해 CL\_GUI\_ALV\_GRID의 이벤트를 호출하기 위한 메소드를 포함하는 클래스를 정의한다.

## 7. 클래스 구현, 8. 이벤트 핸들러 메소드 등록

ALV의 변경된 값을 GT\_MODIFIED\_ROWS 인터널 테이블에 추가하는 로직의 메소드를 구현한다. 즉, 사용자가 ALV에서 값을 변경하게 되면 ALV의 이벤트가 호출되고 이벤트 핸들러 메소드가 변경된 데이터를 인터널 테이블에 저장하게 된다. 물론 이벤트 핸들러 메소드는 등록되어 있어야 한다.

[그림 18-9-2]는 ALV GRID 데이터를 수정하고 저장 버튼을 클릭한후의 결과 화면이다. ALV EDIT 기능을 이해하기 쉽게 ALV GRID의 데이터가 변경된 경우에 대해 소스 스크립트를 최대한 간략하게 축소하였다. 이외 라인을 추가하고 삭제하였을 경우와 같은 여러 이벤트 들에 대해서는 홈페이지 소스 자료실 **Z18\_028** 프로그램을 참고하여 직접 실습하기 바란다. Z18\_027 프로그램을 그대로 복사생성한 후 소스 부분만 복사하여 붙여넣기하면 된다.



▲ 그림 18-9-2. ALV 데이터 변경후 저장 이벤트